Digital Restoration of Damaged Historical Parchment

Kazim Pal

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Engineering
of
University College London.

Department of Computer Science
University College London
June 16, 2015

I, Kazim Pal, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the work.

Abstract

In this thesis we describe the development of a pipeline for digitally restoring damaged historical parchment. The work was carried out in collaboration with London Metropolitan Archives (LMA), who are in possession of an extremely valuable 17th century document called The Great Parchment Book. This book served as the focus of our project and throughout this thesis we demonstrate our methods on its folios. Our aim was to expose the content of the book in a legible form so that it can be properly catalogued and studied.

Our approach begins by acquiring an accurate digitisation of the pages. We have developed our own 3D reconstruction pipeline detailed in Chapter 5 in which each parchment is imaged using a hand-held digital-SLR camera, and the resulting image set is used to generate a high-resolution textured 3D reconstruction of each parchment.

Investigation into methods for flatting the parchments demonstrated an analogy with surface parametrization. Flattening the entire parchment globally with various existing parametrization algorithms is problematic, as discussed in Chapters 4, 6, and 7, since this approach is blind to the distortion undergone by the parchment. We propose two complementary approaches to deal with this issue.

Firstly, exploiting the fact that a reader will only ever inspect a small area of the folio at a given time, we proposed a method for performing local un-distortion of the parchments inside an interactive viewer application. The application, described in Chapter 6, allows a user to browse a parchment folio as the application un-distorts in real-time the area of the parchment currently under inspection. It also allows the user to refer back to the original image set of the parchment to help with resolving ambiguities in the reconstruction and to deal with issues of provenance.

Secondly, we proposed a method for estimating the actual deformation undergone by each parchment when it was damaged by using cues in the text. Since the text was originally written in straight lines and in a roughly uniform script size, we can detect the the variation in text orientation and size and use this information to estimate the deformation. in Chapter 7 we then show how this deformation can be inverted by posing the problem as a Poisson mesh deformation, and solving it in a way that guarantees local injectivity, to generate a globally flattened and undistorted image of each folio. We also show how these images can optionally be colour corrected to remove the shading cues baked into the reconstruction texture, and the discolourations in the parchment itself, to further improve legibility and give a more complete impression that the parchment has been restored.

The methods we have developed have been very well received by London Metropolitan Archives, as well the the larger archival community. We have used the methods to digitise the entire Great Parchment Book, and have demonstrated our global flattening method on eight folios. As of the time of writing of this thesis, our methods are being used to virtually restore all of the remaining folios of the Great Parchment Book. Staff at LMA are also investigating potential future directions by experimenting with other interesting documents in their collections, and are exploring the possibility of setting up a service which would give access to our methods to other archival institutions with similarly damaged documents.

Contents

List of Figures

Chapter 1

Introduction

In this thesis we present our research on the digital restoration of historical parchment. This chapter gives an overview of the background of the project and motivates the problems at hand, summarises the contributions of the thesis, and then explains the structure of the remaining chapters.

1.1 Motivation

Historical documents are an invaluable source of information for historians to gain an understanding of the past. The content of such documents is extremely diverse, ranging from large laws, government records, religious texts, and accounts of battles, to more personal items such as letters, diaries, and personal contracts, which provide a more intimate account of the day-to-day lives of individuals.

As documents age, repeated handling, as well as exposure to light and the air, inevitably cause them to deteriorate, and a great deal of care is taken by the archives, libraries, and museums which store them to preserve them. Sometimes, however, historical documents can deteriorate very severely either through prolonged neglect, or by falling victim to more extreme forms of damage such as fires or floods, leaving them extremely fragile and highly illegible.

Traditional approaches to conserving parchments involve softening them through humidification, followed by the application of force using some apparatus to remove folds and creases. A more thorough description of these methods can be found in Chapters 2 and 3. In cases of extreme damage, however, such traditional methods may be incapable of satisfactorily restoring the

documents and, due to their fragile nature, may even do more harm than good and cause further damage. Even if the conservation is successful the documents will almost inevitably remain extremely fragile.

In this work, we investigate ways in which such extremely damaged parchment documents can be *virtually conserved* using computer graphics, computer vision, and image processing techniques. Digitisation of archival material is already a common practice for documenting the material and making it more easily accessible. However, we would like to move beyond simple digitisation and develop more sophisticated methods which enhance the digitisation to improve legibility of the material and to virtually restore it to a state that more closely resembles its original appearance.

Parchment is an organic material (treated animal skin) and deforms in non-uniform and unpredictable ways when exposed to changes in the environment such as heat or moisture. In Section 2.1 we describe the nature of parchment as a material in greater detail. In our work, we largely focus on undoing the geometric distortions present in such documents. These geometric distortions are detrimental to the legibility of the text since some areas of the document are hidden within creases, and even exposed areas will require some manoeuvring of the parchment to read. Also, publishing images of the text is one of the main goals of the project (which we introduce in Chapter 3, but the 3D shape of the deformed documents often makes it impossible to capture the entire text in a single image. Undoing the deformation and flattening the parchment folio into 2D would allow a single image to capture the entirety of the folio's content.

In our work we apply our methods to a 17th century document called the Great Parchment Book which contains a survey of estates of the Irish County of Londonderry and is therefore an extremely valuable historical resource about the County of Londonderry in the early 17th century. We will introduce the Great Parchment Book in detail in Chapter 3. There are, however, many other documents stored in archives which are in a similar condition and would also benefit from such virtual restoration techniques, including:

- **The Cotton Manuscripts**, damaged by a fire in 1731 at Ashburnham House, Westminster, stored in the British Library [Tite, 1994, Tite, 2003].

- **The Brain**, damaged by a fire in 1731, stored in the British Library (catalogue index Royal MS 9 C X) [Tomalak, 2013].

- **Various medieval Latin parchments**, damaged in the Capitol fire of 1911, stored in the New York State Library [Mercer and Weiss, 2011].

The work described in this thesis is part of a larger project conducted between London Metropolitan Archives and University College London to conserve, transcribe, and digitise the Great Parchment Book, and make it available as a central point of an exhibition in the 2013 commemorations of the 400th anniversary of the building of the city walls in Derry.

1.2 Research Questions and Hypothesis

The overall aim of the work is to develop methods to digitise and virtually restore the folios (individual pages) of the Great Parchment Book, as well as other historical documents with similar types of damage. The most minimal aim is to present the content of the folios in a form that is legible. However, we ideally want the output from our system to be as close as possible to the original state of the parchments before they were damaged, since this is interesting and captivating from a cultural heritage standpoint.

Our hypothesis is that by using an appropriate digitisation method, combined with bespoke visualisation and digital restoration algorithms we can generate images of the parchment folios which would be appropriate for palaeographers to perform a transcription, and which also display the entire content of each folio to facilitate easy dissemination in a way which couldn't be achieved with photographs.

1.3 Contributions

This work makes the following contributions:

- A lightweight and flexible pipeline for digitisation of damaged parchment documents. Specifically our approach captures the shape of the document in 3D since parchment typically deforms into a 3D shape when severely damaged.

- An approach for quantifying the quality of the 3D scans in terms of DPI, since this is the measure used by archives when imaging documents in 2D.

- An interactive application for browsing the 3D scans while always flattening the region being inspected. The application is designed to aid with reading the document, and to mimic the way in which a palaeographer would inspect the physical artefact.

- An interactive method for virtually restoring the digitised parchment based on a sparse set of semi-automatically detected constraints. We show how the size and shape of the text can be used to estimate the deformation undergone by the parchment, and how this deformation can be reverted.

1.4 Structure

In Chapter 2, we discuss previous work on historical document analysis and document rectification, and the related fields of 3D imaging and reconstruction, and surface parametrization, as well as introducing parchment as a writing medium and discussing its nature and the problems that arise when it ages and is exposed to forms of damage. Chapter 3 presents the background of The Great Parchment Book, including its history and the the conservation effort carried out on it prior to our work. In Chapter 4, we carry out a detailed analysis of the problem at hand, how previous approaches can fail, and lay out our proposed solution. Chapters 5, 6, and 7 present the various components of our solution pipeline: the acquisition process, a method of virtually exploring the acquired documents in 3D, and a system for carrying out a virtual restoration of the documents. Chapter 8 summarizes the thesis and discusses potential directions of future work. Appendix A lists a glossary of terms used in this thesis, and Appendix B contains a list of the publications that have been produced as a result of this project.

Chapter 2

Fundamentals and Related Work

In this chapter, we introduce fundamentals of this thesis and discusses related work. Section 2.1 gives an introduction on parchment as a material and writing medium, Section 2.2 presents previous work in the field of document flattening, Section 2.3 gives an overview of historical document analysis unrelated to flattening, Section 2.5 presents state of the art algorithms in surface parameterisation, and Section 2.4 discusses 3D reconstruction methods.

2.1 Parchment

In this section we begin with an explanation of what parchment is, how it is made, and how it differs from other document materials. This is important since it is a key factor in why the problem we are tackling is both difficult and necessary.

Animal skins have been used as a writing media as early as the fourth dynasty of Egypt (c.2550-2450 BC) where documents were written on leather [Diringer, 1982], and Herodotus states (in the 5th century BC) that writing on unhaired sheep and goat skins was commonplace in his time [Thomson and Kite, 2006]. In the 2nd century BC, in Pergamum in Asia Minor, parchment was invented when the process of soaking the skin in a liquor bath and drying it in a stretched state was developed (this process is described in detail in the following Section), and by the 3rd century AD it was the most common writing medium for all types of document [Reed, 1975]. This remained the case until around the 15th century, when it was replaced by paper as the medium of choice, and from then on was only used for luxury manuscripts,

legal documents, archival records, warrants and certificates [Reed, 1975].

2.1.1 Production

The first known account of the production of parchment is found in the Lucca Manuscript written in the 8th century:

> *How parchment is to be prepared: place [the skin] in lime water and leave it there for a few days. Then extend it on a frame and scrape it on both sides with a sharp knife and leave it to dry.*(Codex 490 [Reed, 1975], p. 72).

A more thorough early account is found in the writing of Theophilus, a 12th Century scholar:

> *Take goat skins and stand them in water for a day and a night. Take them and wash them until the water runs clear. Take an entirely new bath and place therein old lime and water mixing well to form a thick cloudy liquor. Place the skins in this, folding them on the flesh side. Move them with a pole two or three times each day, leaving them for eight days (and twice as long in winter). Next you must withdraw the skins and unhair them. Pour off the contents of the bath and repeat the process using the same quantities, placing the skins in the lime liquor and moving them once each day over eight days as before. Ten take them out and wash well until the water runs quite clean. Place them in another bath with clean water and leave them there for two days. Then take them out, attach cords and tie them to the circular frame. Dry, then shave them with a sharp knife after which leave them for two days out of the sun. Moisten with water and rub the flesh side with powdered pumice. After two days wet it again by sprinkling with a little water and fully clean the flesh side with pumice so as to make it quite wet again. Then tighten up the cords, equalise the tension so that the sheet will become permanent. Once the sheets are dry, nothing further remains to be done.*([Reed, 1975], p. 74).

Parchment is typically made from calf, sheep or goat skin, although the skin of other animals could be used. The skin is first soaked in water to clean and

soften it, and then soaked in a lime bath to dissolve the epidermal layer which lines the hair follicles, making the hair easy to remove. When the hair has been removed by scraping it with a knife, it is re-soaked in the lime bath to soften the dermal layer and then washed in water to remove residual lime. The skin is then attached to a frame which it is stretched from all side and is left to dry under tension. It can again be scraped with a sharp knife on both sides to remove any residual hairs and ensure a more uniform thickness. During this drying process, the dissolved collagen settles and sets the skin fibres into the stretched configuration, ensuring they cannot revert to their relaxed state.

2.1.2 Effects of heat and water damage

Subsequent to the liming process, parchment consists mainly of collagen fibres. Woods [Woods, 1995] describes the chemical composition of parchment as "collagen molecules which consist of long chains of amino acids". It is the bonds between these chains which maintain the structure of the fibres. However, when heated in water of a sufficient temperature (65°C for un-limed parchment and 55-60°C for limed parchment), known as the shrinkage temperature, the collagen chains shrink and cause the parchment to shrivel. With further heating, the amino acid chains break down completely and dissolve in the water to form gelatin, causing the parchment to completely harden.

The way in which parchment deforms when damaged in these ways can be very uneven and irregular, due to variations in the thickness and fibre structure, as well as variations in the amount of damage received by different regions of the parchment. This means that very few assumptions can be made about the deformation undergone by a damaged parchment, and thus makes the problem of digitally restoring the parchment difficult as we will discuss in detail in Chapter 4.

2.1.3 Conservation

Many parchments held in libraries and archives are damaged or dirty. In order to store them safely or use them effectively, some conservation treatment is often required.

Conservation of parchment takes on a number of different forms. These

include surface cleaning, repairing holes and tears, and infilling of ink losses. A prerequisite for these treatments, however, is typically to flatten the parchment to remove folds and creases [London Metropolitan Archives, 2013, Woods, 1995].

A standard method of flattening documents used by conservators in the past is to stick it onto another surface (usually a synthetic polyester fabric) in a stretched state using a starch paste [London Metropolitan Archives, 2013, Woods, 1995], and then allowing it to dry under the tension of the adhesive. Woods [Woods, 1995] discusses various problems with this method.

Firstly, the two substrates (document and fabric) need to be flat while the paste dries, otherwise they will not be stuck together well. A common solution to this is to moisten the parchment and flatten it before applying the paste, and then allow it to dry while held down under a weighted board. This method has been described by Clarkson [Clarkson, 1987] as "ignoring its life and spirit altogether...sticking it down...struggling with it and pressing it till it stays down, subjected, beaten and defeated".

Secondly, there is the issue of reversibility. Removing the dried paste from a document usually requires application of water to soften the solidified starch, which can further damage and distort the parchment. Also, some of the starch often penetrates between the parchment fibres, making those regions more rigid and irreversibly changing its internal structure.

A more modern approach is not to completely flatten the document but rather to gently ease open the creases and distortions which cause problems accessing the text. This can be achieved using minimal humidification. That is, leaving the document inside an enclosed space with high humidity to soften it only enough that the creases can be gently eased open with some form of tensioning. [Woods, 1995, London Metropolitan Archives, 2013] This minimal humidification approach has been used on the folios of the Great Parchment Book by conservators at London Metropolitan Archives to improve the accessibility of its text. A detailed account of the conservation procedure used on the book is given in Chapter 3.

2.2 Virtual Document Rectification

Here we examine a number of systems designed for rectifying warped images of documents. Typically such systems deal with simple geometric distortions of a folio such as those caused by the spine of a book, and with regular printed font.

Line-tracing-based methods. Wu and Agam [Wu and Agam, 2002] present a simple method for rectifying a warped image of text based on tracing lines of text through the image and then using these lines to generate a deformation mesh. The method works on binary images with the text segmented from the background, and uses a metric called *local adaptive cumulative projection* to trace the text lines through the image.

If each pixel p has a value g_p equal to zero on text and equal to one elsewhere, the local adaptive cumulative projection, $\phi(p)$ is given by:

$$\phi(p) = \min \phi_\beta(p) \text{ for } \theta - \alpha < \beta < \theta + \alpha, \tag{2.1}$$
$$\text{where } \phi_\beta(p) = \sum_{p \in R(p,\beta)},$$

where θ is the current angle of the text line, α is the maximum angle to search, β is the angle being searched, and $R(p, \beta)$ is a rectangular region beginning at p and aligned with β. The method assumes that the rectangle correctly aligned with the text line will contain the highest density of text pixels.

To begin the line tracing process, a user is asked to define a quadrilateral containing the region of the image to be rectified. The system then searches the left boundary of this quadrilateral for points with high local adaptive cumulative projection, which become the start points for the line tracing. From each of these start points the system iteratively searches for the next point on the line using the same method, until it reaches the boundary of the text region. In order to reduce the possibility of the traced line crossing over into neighbouring lines of text, the maximum angular range is typically lower during the line-tracing stage than when searching for start points. Also the projection in direction β, ϕ_β is modulated by a weight $W(\beta) = 1 + |\tan(\beta - \theta)|/\mu$ to penalize greater

deviations from the current line direction.

Finally, a line-removal step is performed to remove lines which have been traced incorrectly and cross over others. The average orientation in each column, $\bar{\phi}_j$, is computed as $\bar{\phi}_j = \sum_i \phi_{ij}$ where ϕ_{ij} is the orientation of the i^{th} traced line at the j^{th} point. Any line which contains a point which deviates from the average orientation of that column by more than some threshold is removed. Lines which do not reach the right boundary of the text region are also removed.

Once the final set of consistent traced lines has been found, a source mesh is formed by uniformly subdividing each traced line into a number of line-segments and connecting the ends of each line segment across neighbouring lines. The target mesh is formed with the same number of rows as the source mesh. The distance between neighbouring rows in the target mesh is set to the average distance between the corresponding rows in the source mesh, and the columns of the target mesh are spaced uniformly. Finally each cell of the source mesh is warped into the corresponding cell of the target mesh to produce the final rectified image.

Wu and Agam's line-tracing method works well on documents with clear text and good image contrast. However, based on experiments carried out in our research, it is not robust in regions of parchment which are creased or discoloured, or where characters in neighbouring lines overlap due to the handwritten style of the text. As soon as a line of text moves off of the correct baseline it is extremely unlikely to recover. Also, a more complex de-warping procedure would be required to deal with the non-uniform shrinking of the parchment. In Chapter 6 we describe a different method for detecting the direction of text, also based on projections of the text pixels onto different angles, which is more robust in these conditions.

Tian and Narasimhan [Tian and Narasimhan, 2011] propose a system for de-warping of images of text documents by estimating the 3D surface of the document. First they examine the line-structure in the image and generate a 2D distortion grid. Then they generate 3D shape by constraining the grid to be a perspective projection of a 3D parallelogram mesh, and finally unwrap this mesh to produce a rectified image.

Estimation of the 2D distortion grid relies on detecting the horizontal lines of text in the image. This is performed in four stages: First, the scale of the text is estimated. Second, a number of *seed lines* are traced through the image. Third, re-sampling of the seed lines to generate a denser set. Finally, refinement of the lines to accurately localize them to the text.

To estimate the *characteristic scale* of the text in the image, a scale-space pyramid is created by successive down-sampling of the image. At each scale, the *mean gradient magnitude* (MGM) is computed. Tian and Narasimhan observe that which each down-sampling, the scale initially increases since 2D inter-line white-space shrinks more than 1D edges. Then at the scale where letters start to merge together the MGM begins to decrease. This peak is the *characteristic scale* of the image, and the tracing of seed lines is performed on the image down-sampled to this scale.

Tracing of seed lines is performed from a set random points in the image. Starting from a random location x_0, an image patch is extracted centred at x_0. A set of nearby patches, centred at the set of points $\{x_0 + (s\cos\theta_i, s\sin\theta_i)\}_{i=1}^{m}$ where s is the step-size and m is the number of angles to be explored, and choose the patch from this set most similar to the original patch measured using normalized-cross-correlation. Lines are traced from the initial point in both directions until the boundary of the text-region is reached.

Given a set of seed lines $L = \{l_1 ... l_K\}$, Tian and Narasimhan generate a new dense set $L' = \{\tilde{l}_1 ... \tilde{l}_{K'}\}$ by interpolating neighbouring lines in L. This dense set will have many traced line for each actual line of text. To pick exactly one traced line for each text line, the mean-pixel-intensity (MPI) for each traced line is examined for each traced line from the top to the bottom of the image. Local minima of the MPI profile represent a line of text, and local maxima the inter-line white-space. Thus a line $\tilde{l}_i$ from L' is picked if:

$$\text{MPI}(\tilde{l}_{i-1}) < \text{MPI}(\tilde{l}_i) > \text{MPI}(\tilde{l}_{i+1}), \tag{2.2}$$

$$\text{or MPI}(\tilde{l}_{i-1}) > \text{MPI}(\tilde{l}_i) < \text{MPI}(\tilde{l}_{i+1}), \tag{2.3}$$

The resulting lines are then refined to closely fit the top and bottom of each

text line. Given a line l represented by a set of points, each point is vertically displaced to maximise the likelihood that it lies on the text boundary, while also maximising a smoothness term which penalizes sharp changes in the direction of the line.

Next, vertical strokes are detected to estimate the vertical direction in local regions of the text. Tian and Narasimhan try to find the the set of image pixels containing only vertical strokes. They divide the image into a number overlapping patches and perform an optimization based on the gradient magnitudes at each pixel to find only those pixels such that the orientation of the strokes in each patch are approximately equal.

The combination of horizontal and vertical line estimates results in a grid of quadrilaterals estimating the 2D warping of the text. To estimate the 3D shape of the folio from a single such grid, Tian and Narasimhan assume that each cell on the 2D grid corresponds to a parallelogram in 3D space and that camera projection is perspective. They show how, under these assumptions, the positions of each vertex can be found in 3D by minimizing an objective function using a Singular Value Decomposition.

The positions of these vertices defines a 3D parallelogram-mesh, which can be rectified as follows. For each grid cell with corresponding parallelogram P_i, the ratio of the lengths of the sides of P_i is computed, and then the corresponding image patch is warped into a rectangle of the same aspect ratio. Example results are shown in Figure 2.1.

One of the failure cases pointed out in their paper is that on a complicated text layout interspersed with images, the line-tracing algorithm tends to follow straight lines in non-text regions. The parchments we are dealing with often have a non-regular text layout as well as other "features" such as creases, burn marks, etc. which would likely confuse the line-tracing algorithm.

Both Wu and Agam [Wu and Agam, 2002], and Tian and Narasimhan's [Tian and Narasimhan, 2011] methods operate on the idea that best clue for understanding the deformation of a document is by observing the deformation to the text. This idea will be central to the document flattening approaches which we developed, which are described in detail in Chapters 6 and 7. However, both

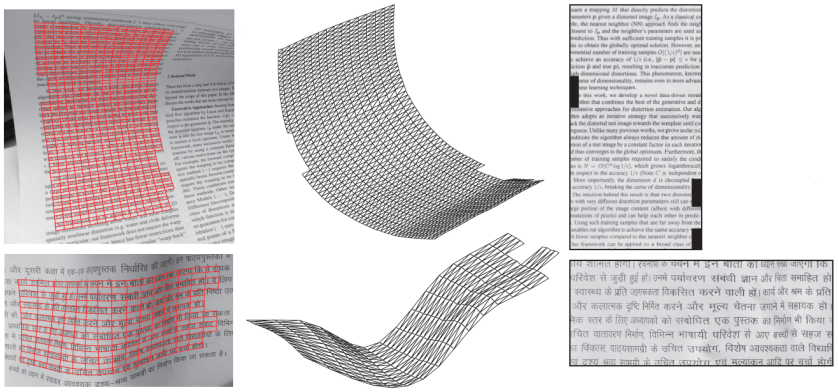

Figure 2.1: Two examples of text rectified using Tian and Narasimhan's method. *Left*: The original images. *Middle*: The estimated 3D surfaces. *Right*: The rectified text. Figure from [Tian and Narasimhan, 2011]. ©2011 IEEE.

of these algorithms make certain assumptions about both the document and the type of deformation it has undergone. They both assume text printed on a clean white page, with regular font and line-spacing, and they also assume that the page has undergone an isometric deformation, which would not necessarily be the case when dealing with parchment. In Chapter 4 we discuss the problem of this isometry assumption in more detail.

3D scanning-based methods. Brown and Seales [Brown and Seales, 2000] present a system for digitizing and virtually flattening arbitrarily warped documents. They begin by acquiring a 3D reconstruction of the document using a structured-light scanner, and then flatten the resulting triangle mesh using a mess-spring simulation.

The 3D reconstruction process consists of a simple structured-light projection setup. A fully calibrated[1] projector and camera are fixed in the scene, with the target object placed in the intersection of their view frustra. A single projector pixel (x, y) is turned on, which illuminates a point on the object. This point is observed in camera pixel (u, v), and so the 3D position of the point is recovered by a stereo reconstruction of these two pixels. This procedure is repeated for every projector pixel to generate a dense set of 3D points representing the object's surface. These points are then triangulated by performing a 2D Delaunay triangulation on the projector's pixels. Since there is a one-to-one

[1]For the purpose of this thesis we use the term "fully calibrated" when referring to a camera or projector to mean one with known intrinsic and extrinsic parameters.

correspondence between the projector pixels and the recovered 3D points, this procedure is equivalent to performing a triangulation on the 3D points.

After the 3D mesh generation the camera is used to photograph the object normally and the resulting image can be used as a texture image. Since each mesh vertex M_i corresponds to a specific camera pixel $\mathbf{u}_i$, the vertices can easily reference the texture by using their corresponding camera pixel as a texture coordinate.

To flatten the triangle mesh, Brown and Seales use a mass-spring simulation in which each vertex of the mesh is treated as a mass and joined by springs to each of its neighbours. An external gravity force is applied to each of the vertices to push them into the $z = 0$ plane, while the spring forces maintain isometry. They use either Euler integration or Runge-Kutta methods to solve the simulation.

Brown and Seales' system produces impressive results considering its simplicity. However the capture process suffers from problems with self-occluding objects since both the camera and projector must be able to see a point on the object to reconstruct it. Such objects would require a number of separate scans to be performed and the resulting meshes registered with each other.Thus it not appropriate for scanning more heavily damaged manuscripts with pronounced folds and creases. The mass-spring approach could also cause unwanted fold-overs when flattening such heavily damaged manuscripts. Finally, the isometry assumption built in to the mass-spring system is not appropriate for completely arbitrarily deformed manuscripts which contain regions that have either shrunk or expanded.

Brown et al. [Brown et al., 2007] use the same structured-light scanning approach of Brown and Seales to acquire a 3D reconstruction of a document, but extend their work by changing the flattening method to use a surface parameterization algorithm, and also add a photometric correction step to remove baked-in shading from the reconstruction's texture.

The mass-spring simulation of Brown and Seales is replaced with the Levy et al.'s Least Squares Conformal Mapping algorithm [Lévy et al., 2002], (which we discuss in Section 2.5) which attempts to embed the vertices of a 3D mesh

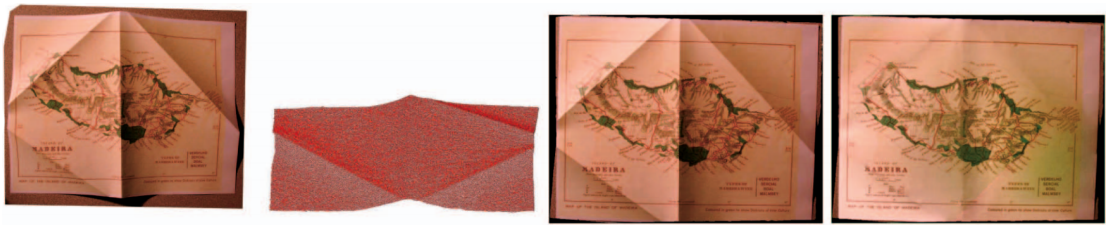

Figure 2.2: *From left to right*: Original image, 3D mesh, geometry corrected, geometry and illumination corrected. Figure from [Brown et al., 2007]. ©2007 IEEE.

into a 2D plane while minimizing the conformal distortion (i.e. the distortion to the angles of the triangles) in a least-squares sense. This approach reduces the risk of introducing fold-overs in the result and also eliminates the problems of instability that physical simulations can suffer from. The conformality assumption, however, still limits the applicability of the method to completely arbitrary deformations, as we will discuss in 4.

Brown et al. also add a photometric correction step into the pipeline to remove shading from the acquired textures, by combining information from the RGB textures and the reconstructed geometry. They note that an edge in the RGB texture either corresponds to a *reflectance edge* in the albedo colour of the manuscript, or a *illumination edge* which are caused by shadows and correspond to discontinuities in the geometry. They remove shading from the flattened image by computing the gradient field of the RGB image, then setting the gradient of pixels located at illumination edges to zero, and integrating back a corrected RGB image by solving a discrete Poisson problem. Some post-processing is finally performed to correct colour bleed-through where genuine albedo edges cross an illumination edge. Brown et al. demonstrate how this correction step can work well for shading caused by creases in the document, but it has no way of handling shading caused by smooth variations in the geometry of document.

Figure 2.2 shows each step of Brown et.al's pipeline, including final results. Brown et al.'s use of mass-spring systems and LSCM for flattening is not suitable for our needs. These approaches make assumptions about the document that are not appropriate for parchment. We will discuss these limitations in detail in Chapter 4. However, the major contribution that they make is the insight that

document flattening is essentially equivalent to surface parameterization. We discuss this in Section 2.5 and build on it throughout this thesis.

Samko et al. [Samko et al., 2014, Samko et al., 2011] present a method for scanning and virtually unrolling scrolled historical documents written on parchment. The scrolls are scanned using an X-ray tomography scanner to produce 3D volumetric data. This data is treated as a set of volumetric slices.

The first step of their method is to segment the parchment from the background. Each slice of the data is filtered using Coherence-Enhancing Diffusion [Weickert, 1999]. This performs smoothing along structure directions in an attempt to fill in missing portions of parchment caused by pores or damage.

Next, Samko et al. segment the parchment from the background using a binary graph-cut optimisation with a shape prior. A standard graph-cut segmentation often leaves many connections between layers of parchment where the adjacent layers are very close. The shape prior takes into account an estimate of the parchment thickness to bias the optimisation against joining adjacent layers.

After the graph-cut segmentation, there are still some false connections between parchment layers in areas where the layers are completely fused together. To solve this problem, Samko et al. separate these areas based on the local features (within a 7×7 moving window) of the parchment's detected borders.

Given an accurate segmentation, the second step is to model the surface. Each parchment folio is separated into two sides by morphological thinning, and then the voxel data is separated by associating each voxel with one side, giving a volumetric representation of each parchment side going several voxels deep into the surface. Next, each tetrahedral meshes are extracted from the voxel data using a constrained Delaunay tetrahedralization [Shewchuk, 2002] approach. The meshes are cleaned to remove isolated vertices, non-manifold vertices, and duplicated elements. Finally it is smoothed using a low-pass filter. The mesh computation procedure is computationally expensive, so Samko et al. only generate it for one end (10 slices) of the parchment and then track it through the rest of the volume by finding correspondences between the surface skeletons in neighbouring slices. Once the entire mesh has been generated, it is flattened

into 2D using the multigrid multidimensional scaling [Bronstein et al., 2006] algorithm, using Dijkstra's algorithm over the mesh vertices to compute surface distances.

Finally, the text is projected onto the flattened 2D surface. The contrast in the X-ray images is provided by the iron in the ink. The brightest pixels in the X-ray data are extracted and projected onto their corresponding triangles in 2D using barycentric coordinates.

Like Brown et al., Samko et al. demonstrate the equivalency between document flattening and surface parameterization, choosing an isometric parameterization method (as opposed to Brown et al.'s choice of a conformal method) to unroll their scrolls. However, this is also unsuitable for our problem because it assumes that the deformation to the document is isometric. As we discuss in Chapter 4, this assumption can break down because of the nature of parchment as a material.

2.3 Digital Analysis of Historical Documents

A number of techniques have been developed to correct aspects of historical document degradation other than geometric distortion, namely fading of the text as the ink degrades, and bleed-through of the text as the document's material degrades.

Multispectral imaging. The Archimedes palimpsest [2] is a 10^{th} century manuscript which contained seven treatises of Archimedes which were later overwritten with the text of a Christian prayer book. Easton et al. [Easton Jr et al., 2003] show how multispectral imaging and processing techniques can be applied to recover the original text.

Experiments showed the original text to be most visible when imaged at short wavelengths and illuminated under longwave ultraviolet (LWUV) light. This is because the original ink is thought to be *redder* than the overwriting ink, and the LWUV illumination causes the parchment to fluoresce which further enhances the contrast. This observation is used to generate pseudocolour images

[2] Palimpsest: A page of a manuscript from which the text has been removed and then overwritten with new text.

of the manuscript which show the overwriting text in black and the original text in colour. Images are collected with a standard digital camera (an Eastman Kodak DCS 760) which has had its anti-aliasing filter removed since the filter blocks infra-red light. The manuscript is then imaged under both tungsten and LWUV light. The red channel of the tungsten image contains the least of the Archimedes text, and the blue channel of the LWUV image contains the most. The red tungsten image channel is then placed into the red channel of the pseudocolour image, and the blue LWUV image channel into the green and blue channels of the pseudocolour image.

Easton et al. also show how a stack of narrowband illumination images can be used to separate different components (parchment, mould, two inks) of the manuscript. To capture these images, they filter the light illuminating the manuscript (in steps of $\Delta\lambda = 50nm$) and using an unfiltered camera. They then show that a constrained least squares algorithm can be applied at each pixel to estimate the ratios of the components at that pixel based on a library of the spectral signatures of each component. The result of this process is a fraction map for each component. That is, an image showing at each pixel the fraction (between 0 and 1) of that pixel made up by that component. These fraction maps can be combined in a similar way as above to generate pseudocolour images of the two texts.

Kim et al. [Kim et al., 2010] proposed an interactive tool that can be used by historians to visualise hyperspectral images of historical documents in a variety of ways. These include spectral selection, spectral similarity analysis, time-varying data analysis, and selective spectral band fusion.

The hyperspectral data is captured using a SEPIA Quantitative Hyper-Spectral Imager (QHSI) device [Klein et al., 2008], which captures each narrow spectral band one at a time by placing a filter in front of the light source to select that spectral band, and then imaging the document under this filtered illumination using a monochromatic camera. Given these hyperspectral images, the software provides a variety of visualisation techniques.

First, the software allows the user to generate RGB renderings of the document under a chosen illumination. The illumination is typically chosen to be

pure white light, but can be set to any spectrum. These simple RGB renderings allow the document to be viewed as it would be naturally seen by the eye.

The user is also able to mark up regions of the document and select specific spectral bands, and the system will compute similarity maps between the mean spectrum of the marked area and other points in the data for the selected bands. This can be useful for segmenting the foreground content from the background of the document and measuring quantities such as the corrosion and ink-bleed severity.

The system can also be used to monitor changes in the document over time due to various factors such as light exposure, changes to humidity, changes to temperature, or combinations of these. If a document is imaged at a number of ageing steps, a slider widget allows the user to move through these steps while seeing histograms of the spectral distribution for the currently selected step. The software can also generate difference images between selected ageing steps to show specifically where in the document changes occur. Kim et al. demonstrate this capability on a document that was artificially aged four times in a variety of ways (light exposure, humidity/temperature change, light exposure + humidity/temperature change, and a control), imaging the document after each ageing exposure.

Finally, it is typically the case that details in the image stack are not distributed evenly over the different spectral ranges, and will often show up in high detail in the invisible range. The software is able to fuse the RGB image with the gradients of this single high-detail spectral band to enhance the content of the RBG image while maintaining its overall appearance. Figure 2.3 shows the result of fusing different regions of an RGB image with different spectral bands which provide the richest gradients for those regions.

Since The Great Parchment Book is not a palimpsest, it was established early on through conversation with London Metropolitan Archives that multispectral imaging would be of limited use for revealing new text. There were, however, a number of folios with particularly faded text which we suspected might benefit from multispectral imaging. However, preliminary imaging experiments by colleagues at LMA showed no improvement.

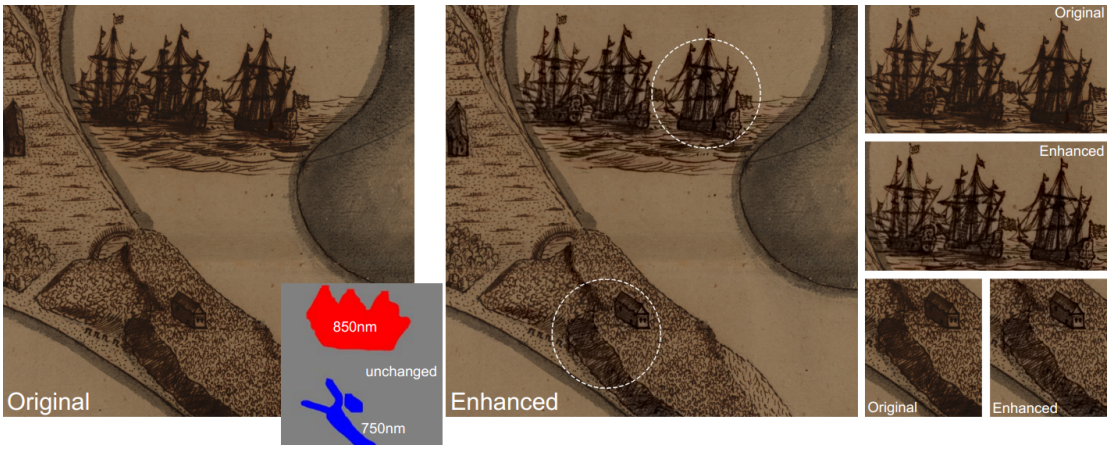

Figure 2.3: Kim et al.'s spectral fusion method. The RGB image is fused with two different spectral bands (750nm and 850nm) to enhance different regions of the image. Figure from [Kim et al., 2010]. ©2010 IEEE.

Ink-bleed removal. Huang et al. [Huang et al., 2010] propose an interactive image editing tool for removing ink-bleed artefacts in images of historical documents. Their tool operates on a pair of images, of the front and reverse sides of a document, and corrects the bleed-though in both images simultaneously. It uses user-provided annotations to learn a classifier which labels pixels of an image as either foreground text, bleed-through text, or background. This classifier is refined in an online way as the user iteratively updates their annotations. Finally, a set of post-processing tools are provided to correct misclassified pixels.

First, the pair of images are aligned to ensure an accurate correspondence between the pixels in each image. Gross misalignments are corrected by the user, and then automatically refined to ensure pixel-level correspondence by optimizing the Normalized-Cross-Correlation between the images.

When this alignment has been established, the user annotates regions of the image as either front text, reverse text, or background, and these annotations are used to compute labeling for the rest of the image using a k-nearest-neighbour method. For each annotated pixel, a feature vector containing the intensities of the front and reverse pixels is collected. To classify an un-labeled pixel its feature vector is extracted and its euclidean distance to each of the labeled pixels is computed. The top k (where k is the square root of the total number of labeled pixels) closest vectors are selected, and the classification is chosen as the majority class within this set. The annotation set can be iteratively updated to refine the

Figure 2.4: *Left*: Documents before ink bleed-through removal. *Right*: Documents after ink bleed-through removal using Huang et al.'s system. Figure from [Huang et al., 2010]. ©2010 IEEE.

classifier.

The classifier will almost always produce errors, especially in areas where the front and reverse text overlap. To correct these errors, the system provides an erase tool to allow the user to remove any pixels misclassified as foreground, and a restore tool to put back ink pixels which were misclassified as background.

Figure 2.4 shows various results of bleed-though corrected using Huang et al.'s system.

Hanasusanto et al. [Hanasusanto et al., 2010] present an automatic approach to ink-bleed removal based on energy minimisation. They adapt the Chan-Vese active contour model [Chan and Vese, 2001] to segment text from a pair of aligned recto-verso document images, by extending it to take into account information from the verso image.

For severely degraded documents, their method will likely generate broken strokes in areas of overlapping recto and verso text. Hanasusanto et al. propose a method for completing broken strokes using the Cahn-Hilliard inpainting energy functional [Bertozzi et al., 2007]. Results of Hanasusanto et al.'s algorithm are shown in Figure 2.5.

Ink bleed-through is not a problem for the majority of the pages of The Great Parchment Book. In Chapter 3, we discuss how a small number of pages have been previously conserved in an overly aggressive manner, causing the parchment to become transparent in certain areas. In these areas there is considerable ink bleed-through, but since this problem is only present in a very small portion of the book, and the text is not rendered completely illegible, it is not considered a primary concern.

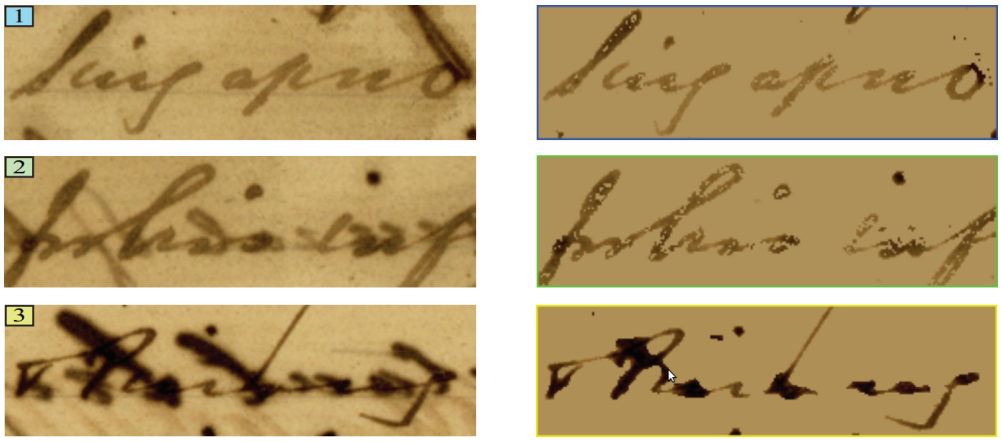

Figure 2.5: *Left*: Documents before ink bleed-through removal. *Right*: Documents after ink bleed-through removal using Hanasusanto et al.'s algorithm. Figure from [Hanasusanto et al., 2010]. ©2010 IEEE.

2.4 3D Reconstruction

Digitally flattening a document requires that we first capture a digital surrogate of the document, to which flattening and restoration algorithms can be applied. For documents with on a small amount of distortion, a 2D image may be sufficient to content of the document at a satisfactory quality. However for more heavily damaged and distorted documents, a 3D representation is required. In Section 2.2 we discussed a number of document flattening methods which begin by generating a 3D reconstruction of the document to be flattened. In this section, we discuss previous work dealing with the general topic of 3D reconstruction of objects.

There are many different approaches to 3D reconstruction. These can be broadly categorised into contact and non-contact approaches. Contact methods acquire the shape of an object by probing it with sensors that make contact with the object. We will not cover this approach since it is unsuitable for use on fragile artefacts that are easily damaged by physical contact. Non-contact methods can be further categorised into active methods, which emit some form of light (or non-visible radiation) to probe the object and detect how the light interacts with the object in order to recover its shape, and passive methods, which analyse reflected ambient radiation to recover the object's shape.

In this section we give and overview of each of these approaches. We will discuss structure-from-motion, a non-contact passive approach, in further detail

since (as we will explain in Chapter 5) this is the approach we use in our work.

2.4.1 Triangulation

Triangulation based scanners, which were introduced briefly in Section 2.2 when discussing Brown and Seales' method [Brown and Seales, 2000], consist of a laser emitter and a camera with known relative positions. A laser dot or laser line is shone onto the object and camera observes where the laser lands on the object's surface, and from this the 3D position of that surface point can be computed. The laser is swept over the entire surface of the object and a model is formed consisting of a dense collection of points sampling the surface [Besl, 1988, Barber et al., 2006, Heritage, 2011].

Triangulation laser scanners were used in the geometry acquisition pipelines of a number of cultural heritage projects. These include the Digital Michelangelo Project [Levoy et al., 2000], the aim of which is to digitise the sculptures of Michelangelo as high quality 3D models and make them available to scholars.

An example of laser triangulation applied to document scanning is Kumar [Kumar et al., 2003], who used a triangulation laser scanner to acquire the geometry of a number of ancient cuneiform tablets. Cuneiform is the world's oldest writing system, where symbols are impressed into damp clay tablets producing a three dimensional text.

Other examples are Brown and Seales [Brown and Seales, 2000] and Brown et al. [Brown et al., 2007] who also use a triangulation laser scanning approach in their document flattening methods, which are discussed in detail in Section 2.2.

2.4.2 Structured light scanning

A similar but faster approach than triangulation is structured light scanning. In this approach, a light pattern consisting of many stripes, or of arbitrary fringes, is projected onto the object. These fringes can be used to reconstruct the object geometry in the same way as a laser stripe, however using a pattern with multiple fringes speeds up the process by allowing many samples to be acquired simultaneously. This, however, requires that each individual strip has to be identified. This is typically achieved by projecting an temporally varying,

alternating stripe pattern onto the object instead of a static stripe pattern. This temporally varying stripe patterns can be chosen (for example Gray codes) so that the illumination pattern for each stripe is unique. Therefore, by observing the stripes over the entire duration of its illumination pattern, they can each be uniquely identified.

Rusinkiewicz et al. [Rusinkiewicz et al., 2002] present a system for real-time 3D model acquisition in which users rotate a small object in-front of a camera and see the most up-to-date version of the acquired model on-screen as the reconstruction is taking place. The key contributions of the paper are the use of structured-light projections to allow for computation of range-images, and the interactive aspect of the system by which the user is able to guide the reconstruction.

The system consists of a real-time reconstruction pipeline composed of the following components:

- Structured-Light Range Scanning, where stripes of light with known patterns are projected onto the object and used to obtain a range image of the object from a known camera position.
- 3D Registration of Range Images, where range-images of the same object from different viewpoints are aligned with each other.
- Merging and Rendering, in which all the aligned range-images are merged into a single representation and rendered on-screen.
- Offline Registration and Surface Reconstruction, where, once the user has acquired scanning data for the entire object, a post-processing step can be run offline to produce a final, higher-quality reconstruction.

The basic structured-light scanning approach consists of observing a pixel for n frames to identify the stripe which illuminated that pixel, however Rusinkiewicz et al. observe that to handle moving objects it is necessary to add frame-to-frame tracking of stripes to accumulate bits of the stripe-code over time, which effectively entails tracking the boundaries between stripes. Rusinkiewicz et al. observe that this permits a more efficient encoding of stripe

information by looking at the on/off status on both sides of a stripe boundary. They develop a stripe sequence which contains 110 unique stripe-boundaries in a four-frame stripe-sequence. This sequence is projected onto the object at a rate of 60 Hz and the resulting illumination is captured by a synchronized camera. This video stream is then used for stripe-tracking and decoding, and a range image is returned.

Each new range image generated by the previous step is aligned with the previous range images using a variation of Besl and McKay's Iterative Closest Point algorithm [Besl and McKay, 1992]. To merge the range images, each one is converted to a 3D point cloud, and points which occupy the same location in a dense voxel-grid are merged. This merged set of points is rendered using a method called *splatting*, where a screen-aligned circle is drawn for each point.

Finally, when the user has determined that enough data has been captured, a final high-quality offline reconstruction is performed on the original range images to generate the final mesh.

The key observation of this work is that by performing a lower-quality reconstruction in real-time and displaying the results on-screen, the user is able to direct the scanning process, fill in holes, and accurately judge when enough data has been collected for the final high-quality reconstruction.

An example of structured light scanning applied to historical documents is the St Chad Gospels [Endres, 2014], where the method was used to capture the 3D geometry of the gospels' pages. The level of distortion in these pages is very minimal so their legibility would not be meaningfully improved by flattening, however the 3D scans allow a reader to have a sense of the state of each folio and allow any feature of the folio to be accurately measured.

2.4.3 Time-of-flight scanning

Time-of-flight (ToF) scanning methods employ a camera system which measures the time-of-flight of a light signal between the camera and the scene and then uses this time to resolve the distance of the scene at each point of the image based on the known speed of light [Iddan and Yahav, 2001]. Since the illumination can be placed next to the camera lens, this approach results in a compact

hardware arrangement compared with other systems based on triangulation and stereoscopic methods which require a certain baseline to be effective. A wide variety of commercial time-of-flight cameras are available, the most famous of which is Microsoft's second generation Kinect sensor. The output of such cameras is a stream of depth images containing a depth value encoded at each pixel, which in itself is not a full 3D reconstruction. In Section 2.4.5 we discuss Izadi et al.'s KinectFusion system [Izadi et al., 2011] which combines these depth images together to form a full 3D representation of the scene or object being imaged.

2.4.4 Multi-view-stereo methods

Stereoscopic systems use two cameras, positioned a small known distance apart, to image an object. By analysing the differences between the two images, a depth map can be computed containing the distance from the camera of the scene at each pixel. Multi-view stereo is a more sophisticated approach is to use an arbitrary number (more than two) of calibrated cameras, and computing a 3D representation (such as depth maps, a point cloud, or a triangle mesh) of the scene.

Deformable Surfaces. Esteban & Schmitt propose an algorithm for reconstructing a 3D surface from a series of calibrated images. Their method is based on a deformable surface model, which is effectively an extension of Kass et al.'s active contour method [Kass et al., 1988] to three dimensions. The algorithm aims to find a surface $S \in \mathcal{R}^3$ which minimizes an energy:

$$E(S) = E_{\text{tex}}(S) + E_{\text{sil}}(S) + E_{\text{int}}(S) \tag{2.4}$$

where E_{tex} is an energy term related to the texture of the object, E_{sil} is an energy related to how well the object matches the silhouettes in the images, and E_{int} is an energy to regularize the surface.

To solve this minimization, Esteban & Schmitt compute the gradient fields of the energies and then introduce a time-series model where at a time k, the

surface S^k is updated as:

$$S^{k+1} = S^k + \Delta t(\nabla E_{\text{tex}}(S^k) + \nabla E_{\text{sil}}(S^k) + \nabla E_{\text{int}}(S^k)) \tag{2.5}$$

The surface is represented as a triangle mesh, initialized to S^0 as the visual hull of the object which is computed from the object's silhouettes in the images. The update step is then run repeatedly until convergence. At each step, each vertex of the current surface is updated according to the forces.

To compute the texture-force F_{tex}, Esteban & Schmitt compute a photoconsistency volume where 3D points, each carrying a correlation score, are stored in an octree structure. To do this they perform mutli-stereo correlation as described in [Hernandez Esteban and Schmitt, 2002] across all images to obtain for each pixel an estimated depth and a correlation score. If the correlation score for a given pixel is greater than some threshold, it is un-projected into 3D world coordinates and stored in the octree.

Once the octree has been filled, the gradient vector flow (GVF) [Xu and Prince, 1998] is computed using a multi-resolution version of the GVF algorithm to generate an everywhere-defined vector field which points towards areas of high correlation and defines F_{tex}.

At a given point $\mathbf{p}$ in space, the silhouette force is defined as follows. First a camera $c(\mathbf{p})$ is selected such the distance between the silhouette S_c in that camera and the projection $\mathcal{P}_c\mathbf{p}$ of $\mathbf{p}$ into the camera is minimal.

Now let d_{VH} be that distance, ie. the smallest signed distance between the silhouette contour and the projection of $\mathbf{p}$:

If d_{VH} is positive, the point is inside the silhouette, and if it negative the point is outside the silhouette. Using d_{VH} alone as the silhouette force would cause the surface to converge on the convex hull.

The final silhouette force is a vector normal to the surface at $\mathbf{p}$ pointing towards the visual hull, whose magnitude is controlled by d_{VH}.

In order to regularize the two external forces, an internal force is defined which attempts to move a given vertex $\mathbf{v}$ to the center of its neighbours. This regularaization favours a smooth solution.

At the k^{th} iteration step, each vertex $\mathbf{v}_i^k$ of the mesh is updated as:

$$\mathbf{v}_i^{k+1} = \mathbf{v}_i^k + \Delta t(F_{tex}^N(\mathbf{v}_i^k) + \beta F_{sil}(\mathbf{v}_i^k) + \gamma F_{int}(\mathbf{v}_i^k)) \tag{2.6}$$

where β and γ are the weights of the silhouette and internal forces relative to the texture force.

After each iteration step, a re-meshing step is performed using edge-collapsing and $\sqrt{3}$ subdivision [Kobbelt, 2000] to maintain a fairly regular triangle mesh and not allow individual triangles to become too large or too small.

Patch-Based Multi-View-Stereo. Furukawa and Ponce [Furukawa and Ponce, 2010] proposed a multi-view stereopsis algorithm for generating a dense point reconstruction of a scene, given a set of calibrated images as input. It is often used after a bundle-adjustment algorithm (discussed in Section 2.4.5) which computes the calibrations of the images.

Their algorithm consists of three steps: *matching*, which generates a sparse set of 3D points, *expansion*, which iteratively expands this set of points to more completely cover the surfaces in the scene, and *filtering*, which removes removes badly reconstructed points by enforcing photoconsistency.

They also propose a polygonal surface reconstruction algorithm for generating a triangle mesh from the point reconstruction, though this is not an essential part of their algorithm and can be replaced with any other meshing algorithm. They suggest Poisson Surface Reconstruction [Kazhdan et al., 2006] as an alternative, which produces comparable results at far lower computational expense. Figure 2.6 shows examples of dense point reconstructions generated by PMVS.

Furukawa and Ponce define a patch p as a rectangle with centre $c(p)$ and unit normal $n(p)$ pointing towards the cameras which observe it. Associated with p is a reference image $R(p)$. The size of p is determined such that its projection into $R(p)$ covers at least a $\mu \times \mu$ patch of pixels (where μ is typically either 5 or 7 in Furakawa and Ponce's experiments). Also associated with p are two sets of images, $S(p)$ and $T(p)$. $S(p)$ is the set of images in which p *should* be visible but may not be recognizable due to specular highlights, moving

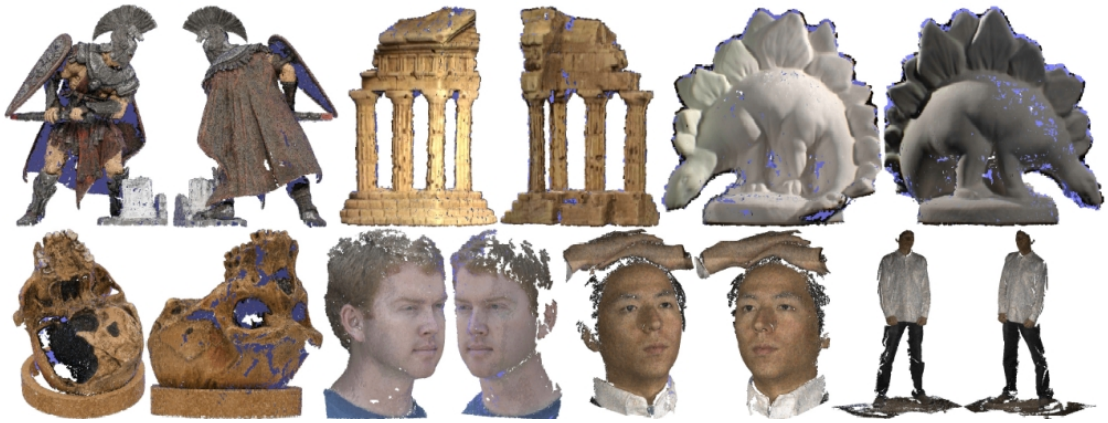

Figure 2.6: Examples of dense point reconstructions using PMVS. Figure from [Furukawa and Ponce, 2010]. ©2010 IEEE.

occluders, etc, and $T(p)$ is the set of images in which p is *actually* visible.

To perform matching, the PMVS algorithm first detects corner and blob features on each of the images using Harris and Difference of Gaussian operators respectively. To ensure the features are spread uniformly over the images, they overlay on each image a regular grid and choose η features per grid cell as the pixels in each grid cell with the maximum responses to the feature detection operators.

To match features across images, each image is considered in turn. For each feature f in cell $C(i,j)$ of image I with camera center O, they collect the set, F, of features f' in other images that are of the same type as f and lie within two pixels of the epipolar line associated with f. For each pair (f, f') a 3D point is triangulated. These points are considered in order of increasing distance from O. At each point a patch p is constructed centred on the point and with a normal vector in the direction of O. The patch that is photoconsistent in at least γ images is selected stored in cell $C(i,j)$ of image I. This process is repeated for all features in all images.

Two patches are considered to be neighbours if they are stored in neighbouring cells of the same image and have similar tangent planes. The expansion step iteratively adds new patches to the neighbours of cells containing existing patches until the patches cover the visible surfaces of the scene. Addition of a new patch during this step is subject to a number of conditions fully detailed by Furakawa and Ponce [Furukawa and Ponce, 2010].

Finally, two filtering steps are performed to remove erroneous patches. The first filter aims at removing patches which lie outside of the actual surface. Given a patch p_0 and the set of patches U which it occludes, p_0 is deemed erroneous if:

$$|T(p_0)|\bar{N}(p_0) < \sum_{p_j \in U} \bar{N}p_j \tag{2.7}$$

where $\bar{N}(p)$ is the average Normalized Cross Correlation (photoconsistency score) of patch p.

The second filter removes patches which lie inside the actual surface. To do this, $T(p_0)$ is recomputed using the depth maps associated with each input image, and p_0 is deemed to be an outlier if $|T(p_0)| < \gamma$.

Traditional two-camera stereo approaches have been used for document imaging by Ulges et al. [Ulges et al., 2004] and Yamashita et al. [Yamashita et al., 2004] who both use a pair of fixed cameras to image the pages of an open book. Due to the book's binding, these pages are curved and the characters on the page are deformed, making them difficult to recognise using character recognition techniques. The stereo images can be used to reconstruct a model of the page's surface, which can be mapped into 2D with the aim of making character recognition more effective.

2.4.5 Structure from motion

Structure from motion is a method, related to multi-view stereo, which takes as input an un-structured, un-calibrated set of overlapping images with a number of image points in correspondence. From these correspondences, the 3D positions of the points are estimated while also estimating the calibration parameters of the images.

Offline methods. We will first examine offline methods, where the image set is captured in its entirety before being processed.

Snavely et al. [Snavely et al., 2006] present a system called Bundler for interactively exploring large unstructured collections of photos in 3D. It consists of two main sections: firstly, a system for 3D reconstruction which takes the unordered image collection and from it estimates camera pose for each of the

images and computes a sparse 3D reconstruction of the scene and, secondly, a photo-explorer interface which can be used to interactively browse the photo collection whilst exploring the 3D reconstruction.

The 3D reconstruction pipeline used by Snavely et al. takes as input an unordered image collection and generates a 3D representation of the scene consisting of:

- A set of points $\mathcal{P} = \{p_1...p_n\}$ each with a 3D position and colour.
- A set of cameras $\mathcal{C} = \{C_1, ..., C_k\}$, each associated with one input image, and with a computed calibration consisting of a rotation matrix R_j, translation t_j, and focal length f_j.
- A mapping, P_{vis}, between points and cameras such that $P_{\text{vis}}(C_j)$ is the set of 3D points visible in camera C_j.

The pipeline, consists of two main phases: Keypoint Detection and Matching, and Structure from Motion.

First, keypoints are detected in every image using Lowe's SIFT keypoint detector [Lowe, 2004]. SIFT provides a local descriptor vector for each keypoint which is used to match keypoints between every pair of images using Arya et al.'s approximate nearest-neighbours algorithm [Arya et al., 1998]. These matches are then used to estimate the fundamental matrix between each pair of images using Hartley and Zisserman's [Hartley and Zisserman, 2004] eight-point-algorithm inside of a RANSAC [Fischler and Bolles, 1981] procedure. Finally, keypoints are organized into *tracks*, which are connected sets of matching keypoints across multiple images, such that a *consistent track* contains at most one keypoint per image. Inconsistent tracks are discarded.

The purpose of this procedure is to recover a set of 3D tracks and camera parameters such that the re-projection error between each track and its corresponding image-space features are minimised. This is a non-linear least-squares minimization of the re-projection error:

$$\sum_{i=1}^{n}\sum_{j=1}^{m} w_{ij}\|q_{ij} - \mathbf{P}(\mathbf{\Theta}_i, p_j)\| \tag{2.8}$$

where p_j is a 3D track, q_{ij} is the corresponding image feature in camera i, $\mathbf{P}(\mathbf{\Theta}, p)$ is a function which projects 3D point p into a camera with parameters $\mathbf{\Theta}$, and w_{ij} is equal to 1 if track j is visible in camera i and equal to 0 otherwise.

Snavely et al. solve this problem in incremental fashion. They start with a single pair of images with a large number of matches, as well as a large baseline so that 3D reconstruction of these matches is well conditioned. They then add the remaining cameras into the optimization one at a time, each time selecting the camera observing the largest number of already-estimated tracks. The intrinsic and extrinsic parameters of this new camera are estimated using the Direct Linear Transformation algorithm from Hartley and Zisserman [Hartley and Zisserman, 2004], and any tracks in the new camera are added into the optimization if they are also observed by at least one camera already in the optimization. The sparse bundle adjustment method of Lourakis and Argyros [Lourakis and Argyros, 2009] is run on this new set of cameras and tracks to minimize the reprojection error function. This process is repeated until all cameras are incorporated.

The output of Snavely et al.'s method is a sparse point-cloud plus calibration parameters for each successfully reconstructed input image. This can be used as input for a dense mulit-view-stereo algorithm such as PMVS, which is discussed in Section 2.4.4.

VisualSFM is an application developed by Wu [Wu, 2011] which provides a graphical user interface for performing 3D reconstruction from a set of images. Wu uses the same approach as Snavely et al.'s Bundler software [Snavely et al., 2006] of feature-matching using SIFT and structure-from-motion using bundle adjustment, but it incorporates Wu's previous work on performing feature-matching on the GPU [Wu, 2007], and multicore bundle adjustment [Wu et al., 2011], making it considerably faster. From a set of input images, it computes camera parameters and a sparse point reconstruction of the scene in the same format as Bundler, and provides an interface by which this output can be used to perform dense reconstruction using Furukawa's PMVS/CMVS tool chain [Furukawa and Ponce, 2010].

Real-time methods. We will now discuss real-time structure-from-motion methods, where the image set is captured, typically as a stream of images from a webcam or similar capture device, while the 3D reconstruction is taking place.

Newcombe and Davison [Newcombe and Davison, 2010] present a method for rapid, dense reconstruction of scenes with a single hand-held camera. The premise of their method is to use a sparse stereo reconstruction of the scene to generate a smooth *base reconstruction* and then warp this base reconstruction into accurate depth maps by optimizing for photometric consistency between bundles of camera frames.

The reconstruction pipeline begins with a set of key-frames (with camera pose) and a sparse point cloud obtained in real-time using Klein and Murray's [Klein and Murray, 2007] Parallel-Tracking-and-Mapping system. From the point cloud they estimate an smooth, continuous surface using the *multi-scale compactly supported radial basis function* method of Ohtake et al. [Ohtake et al., 2003] to fit an implicit surface to the point data. They then extract and polygonize the zero level set of the surface using the method of Bloomenthal [Bloomenthal, 1994] to form an approximate *base mesh* which can now be optimized using *constrained scene flow* to generate an accurate reconstruction.

Newcombe and Davison define a *camera bundle* to consist of a single *reference frame* surrounded by $n \geq 1$ nearby *comparison frames*. Each frame consists of a gray-scale image and a projection matrix $\mathbf{P}$. Camera bundles are selected automatically from the complete set of frames and each one is used to reconstruct a different section of the scene. These separate sections are reconstructed independently and then stitched together to produce the final reconstruction.

Given the base surface and a camera bundle, a ray is back-projected though each pixel, $\mathbf{u}^{\text{ref}}$, in the reference image and intersected with the base surface to give a 3D surface vertex $\mathbf{x}_j$. Each visible point $\mathbf{x}_j$ corresponds to a 2D image point $\mathbf{u}_j^i$ in camera i. Displacing $\mathbf{x}_j$ by some small displacement $\Delta\mathbf{x}_j$ (*scene flow*) gives rise to corresponding displacements, $\Delta\mathbf{u}_j^i$, in image-space. If $\Delta\mathbf{x}_j$ is sufficiently small, $\Delta\mathbf{u}_j^i$ can be approximated using the Jacobian matrix of the projection function of camera i, evaluated at $\mathbf{x}_j$.

By computing the image-space displacements for the comparison cameras

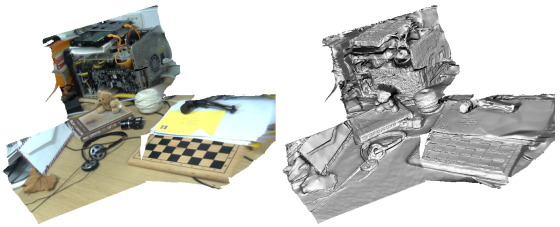

Figure 2.7: *Left*: Textured and *Right*: untextured reconstructions of a desk scene. Figure from [Newcombe and Davison, 2010]. ©2010 IEEE.

in the bundle which minimize the re-projection error, a corresponding object-space displacement can be computed and used to update $\mathbf{x}_j$. These image-space displacements, $\Delta\mathbf{u}_j^i$ are computed for each comparison camera using *view-predictive optical flow*. The reference-frame image is back-projected onto the mesh, and the resulting textured model is rendered into the camera. The optical flow is then computed between this synthesised image and the true image for the camera to give $\Delta\mathbf{u}_j^i$.

The constrained-scene-flow procedure is then iterated until convergence to give a final refined mesh for a single camera bundle. Different models for each camera bundle are then stitched together to form a final reconstruction for the entire scene. An example result of Newcombe and Davison's system is shown in Figure 2.7.

This method gives impressive results but would not be appropriate for capturing a finely detailed mesh or high-quality textures due to the limited resolution of hand-held machine vision cameras. Also, when reconstructing flat surfaces with sharp changes in colour (such as a chequerboard) the algorithm can introduce geometric discontinuities where there should only be discontinuities in texture.

Izadi et al. [Izadi et al., 2011] recently presented a system called Kinect-Fusion which performs real-time 3D reconstruction using Microsoft's Kinect camera. The system is able to build up a 3D model of a scene as it is browsed with a single hand-held Kinect unit. Since subsequent camera frames are highly overlapping, Izadi et al.'s algorithm aligns the two frames and computes the corresponding transformation of the camera pose. With this tracking of the camera pose, the individual depth maps are projected into a single global voxel

volume and fused together by averaging them.

The KinectFusion system is very convenient since the software can be freely downloaded and the Kinect unit is an inexpensive piece of commodity hardware. However, the Kinect sensor has a limited depth and image resolution and the depth-map fusion approach results in blurry looking surface reconstructions lacking in high-frequency details. The approach seems more suited to augmented-reality applications, examples of which are demonstrated in their video, where a highly detailed 3D reconstruction isn't required. For a use case such as ours, where we are interested in capturing the details of the parchment shape as well as a high-resolution texture, the Kinect alone would not be sufficient.

2.4.6 Meshing

The output of many of the above methods is a dense point cloud of the object being acquired. For rendering and processing of the geometry of the object, it is often preferable to represent the object as a triangle mesh. Meshing algorithms compute such triangle meshes from point clouds by fitting a surface to the points.

Kazhdan et al. [Kazhdan et al., 2006] propose a method for reconstructing a 3D surface from a set of oriented points. They formulate the reconstruction as a spatial Poisson problem. Their formulation considers all the points at once, and is therefore robust to noisy data, and can be reduced to a well conditioned sparse linear system.

The insight of their approach regards the relationship between the input point set and the indicator function, χ, of the surface, that is a function defined as 1 inside the surface and 0 outside the surface. The gradient of the indicator function is a vector field that is zero everywhere except at the surface where it is equal to the inward facing normal vector of the surface. The oriented points can therefore be regarded as samples of the gradient of the indicator function.

Given a vector field V defined by the point samples, the indicator function of the surface can be estimated as the scalar field χ whose gradient best approximates V. That is $min_{\chi}\|\nabla\chi - V\|$. This minimization can be formulated into a

standard Poisson problem by applying the divergence operator:

$$\nabla^2 \chi = \nabla V, \tag{2.9}$$

Since the indicator function needs to be computed accurately only near the surface, Kazhdan et.al. implement their solver using an octree structure with each node o having an associate function F_o. They describe in detain how they select F_o such that the vector field V can be expressed as a sum of these per-node functions, and then how the indicator function $\tilde{\chi}$ can be computed. Once $\tilde{\chi}$ has been computed, they extract an isosurface by evaluating $\tilde{\chi}$ at the original sample positions and use the average of these values as the isovalue. The surface extraction is performed using a method which extends the Marching Cubes algorithm [Lorensen and Cline, 1987] to octree structures [Wilhelms and Van Gelder, 1992, Shekhar et al., 1996, Westermann et al., 1999].

Poisson surface reconstruction is a widely uses method for fitting a triangle mesh to point clouds. It is suggested by Furukawa and Ponce [Furukawa and Ponce, 2010] as a method for processing point clouds generated by their PMVS algorithm, and an implementation of it is freely available in MeshLab [Visual Computing Lab of CNR-ISTI, 2005]. For these reasons it would be a convenient and natural choice as a component in our parchment digitisation pipeline.

2.4.7 View dependent texture mapping

View dependent texture mapping is the process by which a texture map is computed for a 3D model of an object or scene from a set of images with corresponding camera calibrations. The images are mapped onto the surface according to the viewpoint that the object is being seen from so that the best images are chosen based on some optimality criteria.

Buehler et al. [Buehler et al., 2001] describe an image based rendering algorithm that combines previous image based rendering methods such as light field rendering [Gortler et al., 1996, Levoy and Hanrahan, 1996] and view-dependent texture-mapping [Debevec et al., 1996]. Unlike typical lumigraph

rendering, which requires images taken at regularly spaced points on a plane, their method can take an unstructured image collection as input, and like view-dependent texture-mapping can also make use of proxy geometry of the scene.

The algorithm is highly flexible in that it will adapt gracefully to whatever input it is given. If the input images are regularly spaced on a plane, and no proxy geometry is provided, the method reduces to typical lumigraph rendering. When provided with highly accurate geometry, the algorithm behaves like regular view-dependent texture-mapping.

Buehler et al. lay out a number of goals they believe an ideal image-based rendering algorithm should meet. These are:

- Use of Geometric Proxies. If scene geometry is available, it should be used to help determine which source rays should be sampled from to reconstruct a desired ray.

- Unstructured Input. The algorithm should accept input images taken in any general configuration.

- Epipole Consistency. If a desired ray passes through the camera center of an of the input images, the algorithm should return the corresponding ray from the source image.

- Minimal Angular Deviation. When blending source rays to reconstruct a desired ray, source rays with similar angles to the desired ray should be preferred.

- Continuity. Two reconstructed rays a small distance apart, intersecting nearby points on the proxy geometry should have correspondingly close colours.

- Resolution Sensitivity. A pixel in an input image is not actually a measure of a single ray, but rather an integral of a number of rays over a small solid angle. This should be taken into account by the algorithm to avoid situations such as rays from a far-away source image being used to reconstruct rays from a close-up viewpoint, as this will result in blurry renderings.

- Equivalent Ray Consistency. The ray along any unbroken line of sight should be reconstructed consistently regardless of the viewpoint position.

- Real-Time Performance. The algorithm should run in real-time.

The algorithm begins by evaluating a *camera blending field* over the image plane which represents the weighting of each camera to reconstruct a given pixel. For each pixel, each camera is assigned a penalty, being the weighted sum of an angular penalty $penalty_{ang}$, a resolution penalty $penalty_{res}$, and a field-of-view penalty $penalty_{fov}$.

When reconstructing a desired ray r_d which intersects the proxy geometry at some point p, we consider each of the source rays r_i from p to the camera centers C_i. The angular penalty of camera i is defined as the angular difference between r_d and r_i:

$$E_{\text{ang}}(i) = \arccos(r_d \dot{r}_i), \tag{2.10}$$

To compute the resolution penalty, there are numerous factors to consider including the difference in distance to the surface of the source view and desired view, surface obliqueness, focal length, and output resolution. Given the projection matrices of each of the source images, the resolution penalty could be accurately computed using the Jacobian of the planar homography relating a source image to the desired view. However, for the sake of efficiency, Buehler et.al. consider only the distance aspect. So given a desired ray with camera center D which intersects the proxy surface at a point p, they compute the resolution penalty of camera i as:

$$E_{\text{res}}(i) = max(0, \|p - C_i\| - \|p - D\|), \tag{2.11}$$

The field of view penalty is used to reject rays which fall outside the field of view of a source image. It is defined as:

$$E_{\text{fov}}(i) = \begin{cases} 0, & \text{if } r_i \text{ is within field of view} \\ \infty, & \text{otherwise} \end{cases} \tag{2.12}$$

Now for a given pixel, camera i has a total penalty:

$$E(i) = \alpha E_{\text{ang}}(i) + \beta E_{\text{res}}(i) + \gamma E_{\text{fov}}(i), \tag{2.13}$$

From these penalties, the weights of each camera are computed. If the penalty of a camera is zero, its blending weight should be infinite relative to the weights of the other cameras, however it is more complex to decide when a cameras weight should drop to zero. Consider first using a global threshold T_{global} such that a camera's weight drops smoothly from a maximum to zero as its penalty increases from zero to T_{global}. Such a method would be unsuitable for unstructured input. To reconstruct a ray with no close source rays, a very large threshold would be required, but where there are many close rays a large threshold would result in the blending of too many cameras and a blurry result.

Instead, Buehler et.al. define at each pixel an adaptive threshold, T_{adaptive}, as the k^{th} largest penalty value in the set of source images. Then the blending weight for camera i is defined as:

$$w(i) = 1 - \frac{E(i)}{T_{\text{adaptive}}}, \tag{2.14}$$

and then normalizing it so that all the weights at a given pixel sum to unity:

$$\tilde{w}(i) = \frac{w(i)}{\sum_{j=1}^{k} w(j)}, \tag{2.15}$$

These weights ensure that only the $k-1$ best cameras (where k is a user-selected parameter) are used to reconstruct a given pixel, and that the blending field change smoothly to avoid visible discontinuities in the rendering. Figure 2.8 shows a rendering of a hallway scene along with the corresponding camera-blending-field. To render a given viewpoint, the blending field is evaluated at a number of sample points in the image plane and then interpolated over the entire image plane. Sample points are selected first by projecting both the vertices of the proxy geometry and the camera centers of each of the source images into the desired view, and then finally adding a regular grid of points. These points are then triangulated using a constrained Delaunay triangulation

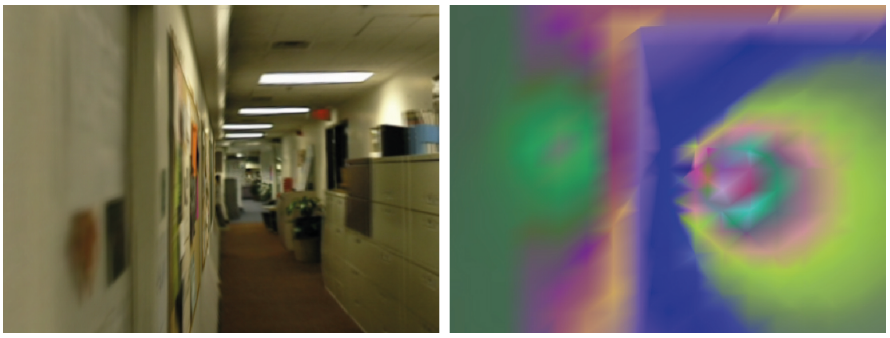

Figure 2.8: A rending and corresponding camera-blending-field generated using Unstructured Lumigraph Rendering. Different colours in the blending field represent different input cameras. Figure from [Buehler et al., 2001]. ©2001 ACM.

where the edges of the proxy geometry and the edges the regular grid are added as constraints on the triangulation. Then the triangles are rendered once for each source input image, with the source image as the texture and with the computed blending weights for the sample points in the alpha channel.

Buehler et al.'s approach does a good job at making the most of sparse or incomplete input, however in some cases the approach can fail and introduce very noticeable artefacts. For example, a foreground object which is not represented in the proxy geometry will appear to move suddenly as the camera used to render it changes. Also, if the input cameras are more densely distributed in one direction than another, which can cause the rendering to be blurry in the direction of dense sampling and to contain discontinuities in the direction of sparse sampling [Davis et al., 2012]

We do not make use of the rendering algorithm directly. However, as we describe in Chapter 5, we adapt Buehler et al.'s weighting and interpolation scheme to blend together a set of images of a parchment folio to generate smooth texture maps for 3D reconstruction of the folio.

2.5 Surface Parametrization

A surface parametrization is a bijective mapping C of a parameter domain $D \subseteq \mathbb{R}^2$ onto a 3D surface $S \subseteq \mathbb{R}^3$:

$$C : D \to S, \tag{2.16}$$

$$C(u,v) = (x(u,v), y(u,v), z(u,v)), \tag{2.17}$$

$$C^{-1}(x,y,z) = (u,v), \tag{2.18}$$

Flattening a document is equivalent to parameterizing the surface, since this finds a way to embed the document surface in 2D. The link between document flattening and surface parameterization has already been explored by Brown et al. [Brown et al., 2007] and Samko et al. [Samko et al., 2014], whose methods were discussed in Section 2.2.

Tenenbaum et al. [Tenenbaum et al., 2000] present an algorithm for computing an isometric embedding of high-dimensional data into a low-dimensional space. It is based on classic Multidimensional Scaling (MDS) [Kruskal, 1964] but incorporates geodesic distances on a weighted graph in order to learn the geometry of the data. Consider the *Swiss Roll* dataset illustrated in Figure 2.9. Points that are close together in the three-dimensional space may actually be far apart on the underlying manifold. By computing the geodesic distances over the manifold between the points, the Isomap algorithm is able to detect the intrinsic two-dimensionality of the points. The algorithm proceeds in three steps.

First, a neighbourhood graph is computed. If no connectivity information is provided between the points, Tenenbaum et al. propose two ways to determine the connectivity. The first method is to deem two points i and j to be neighbours if the distance between them is less than some threshold value. The second method is to set i as the neighbour of j if i is one of j's K nearest neighbours. However, if the surface being parameterized is a triangle mesh, the neighbourhood graph is already defined by the connectivity of the mesh.

Next, the geodesic distances in the graph are estimated by computing the shortest path through the graph between all pairs of points using Dijkstra's algo-

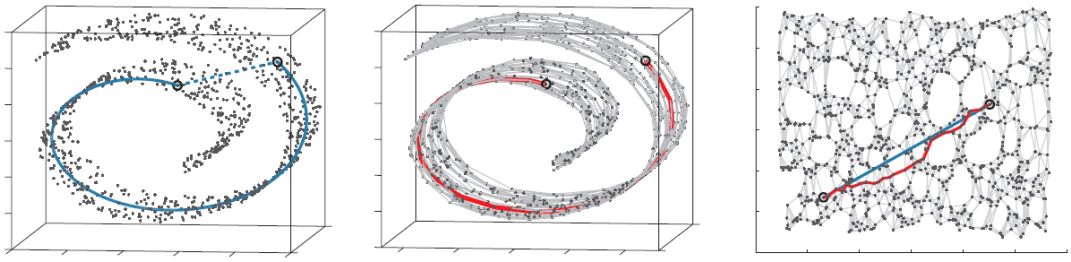

Figure 2.9: The Isomap algorithm applied to the Swiss Roll dataset. (a) The dashed blue line (three-dimensional Euclidean distance) does not accurately represent the distance between the two points on the manifold (solid blue line). (b) The neighbourhood graph is constructed and the geodesic distance is estimated as the shortest path between the two points (red line). (c) The two-dimensional embedding recovered by Isomap. The euclidean distance (blue line) in this space between two points is a good estimate of their true geodesic distance on the manifold. From [Tenenbaum et al., 2000]. Reprinted with permission from AAAS.

rithm [Dijkstra, 1959]. If the points in the graph densely sample the underlying surface, this will provide good approximations to the true geodesic distances.

Finally, the an embedding of the data is found in the lower-dimensional space by applying classical Multidimensional Scaling to the geodesic distances. It finds the coordinates $\mathbf{y}_i$ in the d-dimensional space, Y, by minimizing the cost function:

$$E = \|\tau(D_G) - \tau(D_Y)\|, \tag{2.19}$$

where D_G is the matrix of geodesic distances, D_Y is the matrix of euclidean distances between each pair $(\mathbf{y}_i, \mathbf{y}_j)$ in the space Y, and τ is a function which converts the distance matrices to matrices of inner products.

Tenenbaum et al. demonstrate their algorithm on the Swiss Roll dataset, shown in Figure 2.9, where a manifold in three-dimensional space is unwrapped into a two-dimensional plane.

The primary bottleneck of classical MDS (and hence of the Isomap algorithm) is computing the eigendecomposition of the $N \times N$ distance matrix D (where N is the number of points). The computational cost of this computation grows quadratically with N. Silva & Tenenbaum propose a variation of MDS called Landmark-MDS (or L-MDS) [V. de Silva, 2004], which has a cost that is

essentially linear in N. It takes as input an $n \times N$ matrix (a sub-matrix of D) which contains the distances between the N data points and a subset of the data points of size n with $n << N$. These n points are referred to as landmark points.

The L-MDS algorithm has two steps. First, classical MDS is performed on the n landmark points to find their d-dimensional embedding. Next the embedding of the remaining points are computed based on their distances to the landmarks as follows:

Let Δ_n be an $n \times n$ matrix of the pairwise squared distances between the landmark points, and let $\mathbf{ffi}_1 ... \mathbf{ffi}_n$ be the columns of Δ_n, so $\mathbf{ffi}_i$ is a vector of squared distances between the i^{th} landmark point and the other landmark points. Now let $\mathbf{ffi}_\mu$ be the mean $\mathbf{ffi}_\mu = (\mathbf{ffi}_1 + ... + \mathbf{ffi}_n)/n$.

Then letting L_k be the matrix $[\mathbf{l}_1 ... \mathbf{l}_n]$ where $\mathbf{l}_i$ is the embedding of the i^{th} landmark, compute the pseudoinverse transpose matrix, $L_k^\#$, of L_k. In their paper, Silva & Tenenbaum describe a closed-form expression for this matrix based on the original eigenvectors and eigenvalues computed when performing classic MDS on the landmarks.

Finally, letting $\mathbf{ffi}_\alpha$ be the vector of squared distances between a data point α and the landmarks, the embedding $\mathbf{x}_\alpha$, of α is given by:

$$\mathbf{x}_\alpha = -\frac{1}{2} L_k^\# (\mathbf{ffi}_\alpha - \mathbf{ffi}_\mu), \tag{2.20}$$

Silva & Tenenbaum show that to perform an embedding in a k-dimensional space, at least $k+1$ landmark points are required. If the data is free of noise, $k+1$ landmarks is sufficient to produce the same results as classical MDS, however in the presence of noise more landmarks are required for the algorithm to perform robustly. Also, the landmarks should be distributed uniformly over the data and span all k dimensions of the manifold. Landmarks that are nearly co-linear, for example will cause poor results.

Lévy et al. [Lévy et al., 2002] present a method for generating a texture atlas for a triangle mesh by first decomposing the mesh into sub-meshes, each homeomorphic to a disc, mapping each of these sub-meshes conformally into uv space, and finally packing these separate uv-parametrizations into a texture

atlas. Since a parchment folio is homeomorphic to a disc to begin with, we will focus only on the mapping step and ignore the decomposition and packing steps as they are not relevant to our research.

A mapping from a (u, v) domain to a 3D surfaces, $\mathcal{X} : \mathbb{R}^2 \rightarrow \mathbb{R}^3$, is said to be conformal if for a given (u, v), the iso-u and iso-v curves on the surface are orthonormal. Letting $N(u, v)$ denote the unit normal of the surface at (u, v), we can write that:

$$N(u,v) \times \frac{\partial \mathcal{X}}{\partial u}(u,v) = \frac{\partial \mathcal{X}}{\partial v}(u,v), \tag{2.21}$$

Lévy et al. state that this equation cannot be enforced strictly since we are now dealing with a triangulation where triangle edges need to be mapped to straight lines, so instead they minimize a conformality condition:

$$C(T) = |\frac{\partial \mathcal{U}}{\partial x} + i\frac{\partial \mathcal{U}}{\partial y}|^2 A_T, \tag{2.22}$$

where A_T is the area of triangle T. This is summed over the entire mesh $\mathcal{T}$ to give the criterion that must be minimized:

$$C(\mathcal{T}) = \sum_{T \in \mathcal{T}} C(T), \tag{2.23}$$

Lévy et al. then go on to show that, given the full set of vertices in the mesh, some of which are categorized as *free* and others as *pinned* to a pre-defined location in (u, v) space, this minimization can be formulated as:

$$C(\mathbf{x}) = \|A\mathbf{x} - b\|^2, \tag{2.24}$$

where A, b, and $\mathbf{x}$ are matrices and vectors formulated from the coordinates of the vertices. Their exact formulation can be found in [Lévy et al., 2002]. The free variables of the optimization can then be found by solving $A\mathbf{x} = b$ for $\mathbf{x}$ in a least-squares sense.

Sheffer and Sturler [Sheffer and de Sturler, 2001] present a method for computing a planar triangulation of a 3D triangle-mesh based on a constrained optimization based on the angles of the mesh.

The constrained optimization step involves minimizing an objective function over the vector, **ff**, of all angles in the flattened mesh:

$$F(\mathbf{ff}) = \sum_{i=1}^{P} \sum_{j=1}^{3} w_i^j (\alpha_i^j - \phi_i^j)^2, \tag{2.25}$$

where P is the number of triangles in the mesh, $\alpha_i^j$are the angles of the i^{th} triangle in the flat mesh, ϕ_i^j is the optimal angle for α_i^j based on the geometry of the original mesh, and w_i^j is a weighting term.

To ensure the mesh resulting from this optimization is valid, Sheffer and Sturler introduce additional constraints, which they incorporate into the objective function. This augmented objective function, the formulation for which can be found in Sheffer and Sturler's paper, is minimized using Newton's method, using the optimal angles θ_i^j as the initial guesses.

This procedure generates an optimal vector of angles, $\hat{\mathbf{ff}}$, in the flattened mesh. Vertices are placed in the 2D plane by iterating over the edges in the mesh and positioning the vertices of each edge to satisfy the optimal angles, resulting in a 2D parametrization of the 3D mesh.

These two methods use a deformation energy based on maintaining the conformality of the triangle mesh. Sorkine and Alexa [Sorkine and Alexa, 2007] define a different mesh deformation energy which allows a triangle mesh to be deformed by a number of constraints on the positions of its vertices, while attempting to maintain rigidity across the rest of the mesh.

Given a vertex, $\mathbf{v}_i$, and all the vertices, $\mathbf{v}_j$, of the triangles incident on $\mathbf{v}_i$, and transformed versions of these vertices, $\mathbf{v}_i'$ and $\mathbf{v}_j'$, we can say that the transformation is rigid if there exists some rotation matrix $\mathbf{R}_i$ such that:

$$\mathbf{v}_j' - \mathbf{v}_i' = \mathbf{R}_i(\mathbf{v}_j - \mathbf{v}_i), \forall j \in \mathcal{N}(i), \tag{2.26}$$

Therefore, the total rigidity energy $E(S')$ of a transformed surface can be expressed as:

$$E(S') = \sum_{i=1}^{n} \sum_{j \in \mathcal{N}(i)} w_{ij} |(\mathbf{v}_j' - \mathbf{v}_i') - \mathbf{R}_i(\mathbf{v}_j - \mathbf{v}_i)|^2, \tag{2.27}$$

where the weights w_{ij} are computed using the cotangent weight formulation to account for irregularly shaped triangles and make the surface deformation as independent as possible from the specific mesh discretisation.

Minimizing this energy to solve for the vertex positions $\mathbf{v}_i$ results in the following sparse linear system for each vertex $\mathbf{v}_i$:

$$\sum_{j \in \mathcal{N}(i)} w_{ij}(\mathbf{v}'_j - \mathbf{v}'_i) = \sum_{j \in \mathcal{N}(i)} \frac{w_{ij}}{2}(\mathbf{R}_i + \mathbf{R}_j)(\mathbf{v}_j - \mathbf{v}_i), \tag{2.28}$$

which can be written in matrix form as:

$$\mathbf{L}\mathbf{p}' = \mathbf{b}, \tag{2.29}$$

where $\mathbf{L}$ is the well known discrete Laplace-Beltrami operator [Reuter et al., 2009], and $\mathbf{b}$ is a vector whose elements are computed from the right hand side of the above linear system. Since $\mathbf{L}$ is a symmetric positive definite matrix, the system can be solved efficiently using a sparse Cholesky factorization solver [Horn and Johnson, 1986].

Many mesh deformation algorithms can often introduce inverted triangles in the case of extreme deformations. Schüller et al. [Schüller et al., 2013] propose a method for modifying any existing deformation energy, such as those used for surface parameterization, to guarantee that the deformation results in a locally injective mapping - that is, one which contains no triangle flips. They do this by introducing a *barrier term* into the deformation energy, which grows as a triangle is about to invert.

Schüller et al. begin with a function $\lambda_j \mathbf{v}$ which measures the area of a triangle j under a given vertex configuration $\mathbf{v}$. From this, they define the constraint term:

$$c_j \mathbf{v} = \lambda_j(\mathbf{v}) - \epsilon, \tag{2.30}$$

where ϵ is a small constant used to account for numerical inaccuracies. By preventing this term from ever becoming negative, local injectivity is guaranteed.

The barrier function ϕ_j is then defined based on a cubic polynomial term g_j:

$$g_j(x) = \frac{1}{s_j^3}x^3 - \frac{3}{s_j^2}x^2 + \frac{3}{s_j}x, \tag{2.31}$$

as:

$$\phi_j(x) = \begin{cases} \infty & x \leq 0, \\ \frac{1}{g_j(x)} - 1 & 0 < x < s_j, \\ 0 & s \geq s_j, \end{cases} \tag{2.32}$$

The constant s_j determines the amount by which a triangle can shrink before the barrier term begins to intervene. Schüller et al. suggest that values ranging from $0.1 - 1.0$ typically work well based on their experiments.

Now, given an energy term $\mathbf{E}(\mathbf{v})$, a constraint matrix $\mathbf{C}$, and constraint target vector $\mathbf{d}$, using the barrier function yields the following optimization problem:

$$\underset{\mathbf{v}}{\operatorname{argmin}}\, \mathbf{E}(\mathbf{v}) + \alpha|\mathbf{C}\mathbf{v} - \mathbf{d}| + \beta \sum_j \phi_j(c_j(\mathbf{v})), \tag{2.33}$$

where α and β are parameters controlling the strength of the constraint and barrier terms respectively.

The optimization is solved using a variation on the Levenberg-Marquardt algorithm [Moré, 1978]. In cases of extreme deformation, Schüller et al. use a sub-stepping approach where the target destinations for constrained vertices are moved to intermediate positions between their final target position and that vertex's current position. The optimization is solved with these intermediate constraint targets, and then the intermediate targets are moved forward towards their final positions. This process is repeated until the constraint targets are in their final locations. Results of Schüller et al.'s method are show in Figure 2.10. This guarantee of local injectivity is useful for a document flattening problem since real materials (such as parchment) cannot compress to a zero or negative area.

Bommes et al. [Bommes et al., 2009] present a method which converts a given triangle mesh into a quad mesh.

A smooth, symmetric cross-field is generated on the mesh to satisfy a set of constraints which capture the geometric structure of the surface. The

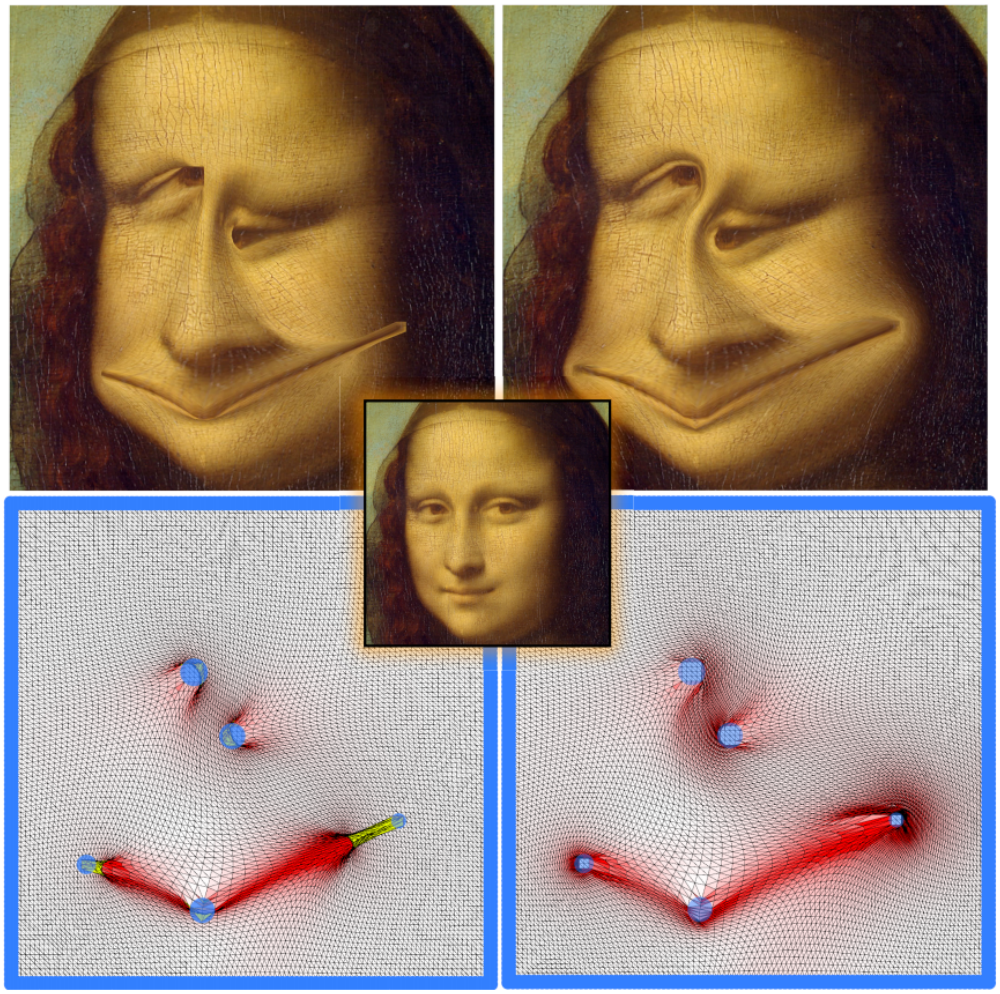

Figure 2.10: *Left*: The standard ARAP energy is used to deform a textured regular mesh. The extreme deformation causes inversion of some triangles (yellow), resulting in the artefacts visible in the textured image. *Right*: Adapting the ARAP energy to be locally injective prevents these triangles from inverting and removes the artefacts. Figure courtesy of [Schüller et al., 2013].

constraints are generated by finding parabolic regions, since these have a well-defined orientation. These regions can be found by measuring the relative anisotropy of the principal curvatures. To compute these curvatures, Bommes et al. compute the shape operator [Cohen-Steiner and Morvan, 2003] at each point on the surface. For points with a valid shape operator, with relative anisotropy greater than some minimum threshold, the add a directional constraint since these are points of salient curvature direction.

A smooth cross-field over the entire surface which satisfies these constraints, can be found by minimizing the sum of all squared differences between the cross-field angles of neighbouring triangles, subject to the directional constraints.

To perform this minimization, Bommes et al. introduce a greedy mixed-integer solver.

Next Bommes et al. show how the mesh can be parameterized in a way that is locally oriented to the previously computed cross-field. They assign a (u, v) parameter to each vertex such that the gradients of u and v match the cross-field orientation, yielding a local orientation energy for each triangle T:

$$E_T = \|h\nabla u - \mathbf{u}_T\|^2 + \|h\nabla v - \mathbf{v}_T\|^2$$

where h is a scaling parameter which controls edge lengths in the output mesh and $\mathbf{u}_T$ and $\mathbf{v}_T$ are the orthogonal directions of the cross-field in T. Then, a global orientation energy is defined as:

$$E_{\text{orient}} = \sum_T \omega(T) E_T \text{Area}(T)$$

where ω is a per-triangle weighting term which penalises high distortion and triangle-flips. Bommes et al. minimise this energy by solving the sparse linear system which sets each of the energy's partial derivatives to zero.

Praun et al. [Praun et al., 2000] propose a method for texturing an arbitrary triangle mesh using an example 2D texture. The method relies on the observation that any manifold surface may be locally parameterized into 2D. Patches of the texture are pasted onto regions of the mesh that can be easily parameterized. By repeating this process, the entire mesh is covered with texture patches, which Praun et al. refer to as a lapped texture.

The orientation and scale of the texture at each point of the mesh can be controlled by the user, who can specify a tangential vector field on the mesh surface. The user is able to specify vectors on a select number of mesh triangles, which are then interpolated over the remaining faces using Gaussian radial basis functions. The user also has control over the weight and extent of the basis functions.

Next the surface is divided up into a set of surface patches, onto which the texture is pasted. A random point is chosen on a face that hasn't yet been

textured. A patch is grown around this seed point by adding new triangles in order of distance from the seed point. The patch is grown over an edge only if the new resulting patch is homeomorphic to a disc, the edge is still inside the texture patch, and the distortion is not excessive. The parameterization, $\phi : \mathbb{R}^3 \rightarrow \mathbb{R}^2$ is initialized such that the seed point maps to the center of the texture patch, and the local basis $(\mathbf{S}, \mathbf{T})$ of the seed point's triangle is aligned to the texture space axes $(\hat{\mathbf{s}}, \hat{\mathbf{t}})$.

The parameterization is then optimized to locally match the vector field defined on the surface. For each face $f = \mathbf{A}, \mathbf{B}, \mathbf{C}$, the up-vector $\mathbf{T}$ can be expressed in barycentric coordinates as:

$$\mathbf{T} = \alpha\mathbf{A} + \beta\mathbf{B} + \gamma\mathbf{C}, \text{, where } \alpha + \beta + \gamma = 0$$

Since the map ϕ is linear over the face, the image of $\mathbf{T}$ in the texture space is a linear combination of the parameterizations of the face's vertices. A difference vector $\mathbf{d}_T$ can be defined as:

$$\mathbf{d}_T = \alpha\phi(\mathbf{A}) + \beta\phi(\mathbf{B}) + \gamma\phi(\mathbf{C}) - \hat{\mathbf{t}}$$

and similarly for $\mathbf{d}_S$. The optimization of ϕ can be found by minimizing the energy function:

$$\sum_f \|\mathbf{d}_S\|^2 + \|\mathbf{d}_T\|^2$$

which can be done by solving a sparse linear system, which Praun et al. do using a conjugate-gradient iterative solver.

The parameterized textures can either be stored in a traditional texture atlas, or can be composited and rendered at runtime using the hardware graphics pipeline.

These surface parameterization methods by Bommes et al. [Bommes et al., 2009] and Praun et al. [Praun et al., 2000] show how directional constraints can be used to influence the parameterized surface. In Chapter 7 we will show how a similar approach can be used to globally flatten parchment folios.

Chapter 3

The Great Parchment Book

In this chapter, we introduce the Great Parchment Book in detail. We explain the history of the book, and motivate why it is so valuable to historians, and then give an overview of the physical conservation work which has been conducted on the book by conservators at London Metropolitan Archives (LMA).

3.1 History of the Great Parchment Book

The Great Parchment Book of the Honourable The Irish Society is a survey of the estates in the Northern Irish county of Derry managed by the City of London through the Irish Society and the City of London livery companies. It was compiled in 1639 by a Commission under King Charles I and is a very important source of information about the City of London's role in the Protestant colonisation [Curl, 2000, Moody, 1939].

The Irish Society was formed in 1609 by the City of London Corporation, to manage the estates in Derry which King James I had obliged it to take on, as part of the plantation of Ulster. The Irish Society took charge of the overall management of the estates, with direct control of the city of Londonderry.

The Great Parchment Book was compiled in the late 1630s when Charles I claimed the estates after a court ruled that the Londoners had not fulfilled their obligations of plantation. Sir Ralph Whitfield, serjeant-at-law, Thomas Fotherley, gentleman of the King's Privy Chamber, Bramhall (along with a clerk and two assistants) carried out the settlement between April and October 1639. All of the new contracts were collected in a "great parchment booke" [Moody, 1939], which later came into the possession of the Irish Society and was kept in the

Figure 3.1: Folios of The Great Parchment Book stored in a box. (Reproduced with the permission of The Honourable The Irish Society and The City of London Corporation, London Metropolitan Archives)

Guildhall in the City of London.

In February 1786, a fire in the Chamber of London at the Guildhall destroyed most of the early records of the Irish Society. The Great Parchment Book survived but was damaged so severely by the fire that it has been completely unavailable to researchers ever since, as noted by Moody [Moody, 1939] and Curl [Curl, 2000].

The manuscript has since been stored at London Metropolitan Archives (LMA reference CLA/049/EM/02/018.) The manuscript contains information that is extremely valuable to historians regarding landholding and population in Derry in the 17th century around the time of the Protestant colonisation, as well as the county's relationship with the City of London. The manuscript consists of 165 folios of varying levels of damage. It is unknown how many folios were in the original book before the fire, and the ordering is also unknown. The folios have been grouped together by conservators at LMA according to the livery company which they seem to refer to (for example, all the folios referring to the fishmongers company in one group), although they have told us that this does not constitute a strict ordering and it is sometimes ambiguous how a folio should be grouped. Figure 3.1 shows a how the folios of the book were stored prior to grouping and re-packaging by LMA.

In 2010 a partnership was formed between London Metropolitan Archives,

the Department of Computer Science and the Centre for Digital Humanties at University College London (UCL). The goal of this partnership was to attempt to make the Great Parchment Book available as a central component of an exhibition being held in Derry in 2013 to commemorate the 400th anniversary of the building of the city walls, and also to publish both a transcription and images of the book online.

3.2 Physical Conservation of the Great Parchment Book

Prior to the work described in this thesis, conservators at London Metropolitan Archives carried out some physical conservation methods designed to make the subsequent digital imaging more effective [London Metropolitan Archives, 2013]. The first step of the conservation process was to assess the different types of damage present in the book and define a measure of the damage to each folio. The damage to the parchments was found to take the following major forms:

Planar Distortion. This is the most obvious form of damage. The denaturing of the parchment fibres when exposed to heat causes the parchment to crease and shrivel in very unpredictable ways and results in the "poppadom-like" appearance of the folios.

Shrinking. The heat damage also causes the parchment fibres to contract resulting in shrinking of the text. Figure 3.2 shows an example of this shrinking where a letter P can be seen to have shrunk to less than half of its original size.

Tears. Some areas of the parchment, typically on the edges of individual folios, contain tears. In some cases, there is no loss of material and the the tear could, in principle, be joined up. In other cases, however, the tears are more severe and have become holes where parchment has been lost.

Calcite Formation. Calcite is a mineral which is naturally present within the structure of parchment, but moves to the surface when the parchment is subject to moisture and high temperatures. It is visible as a white powder-like substance, as can be seen in Figure 3.3.

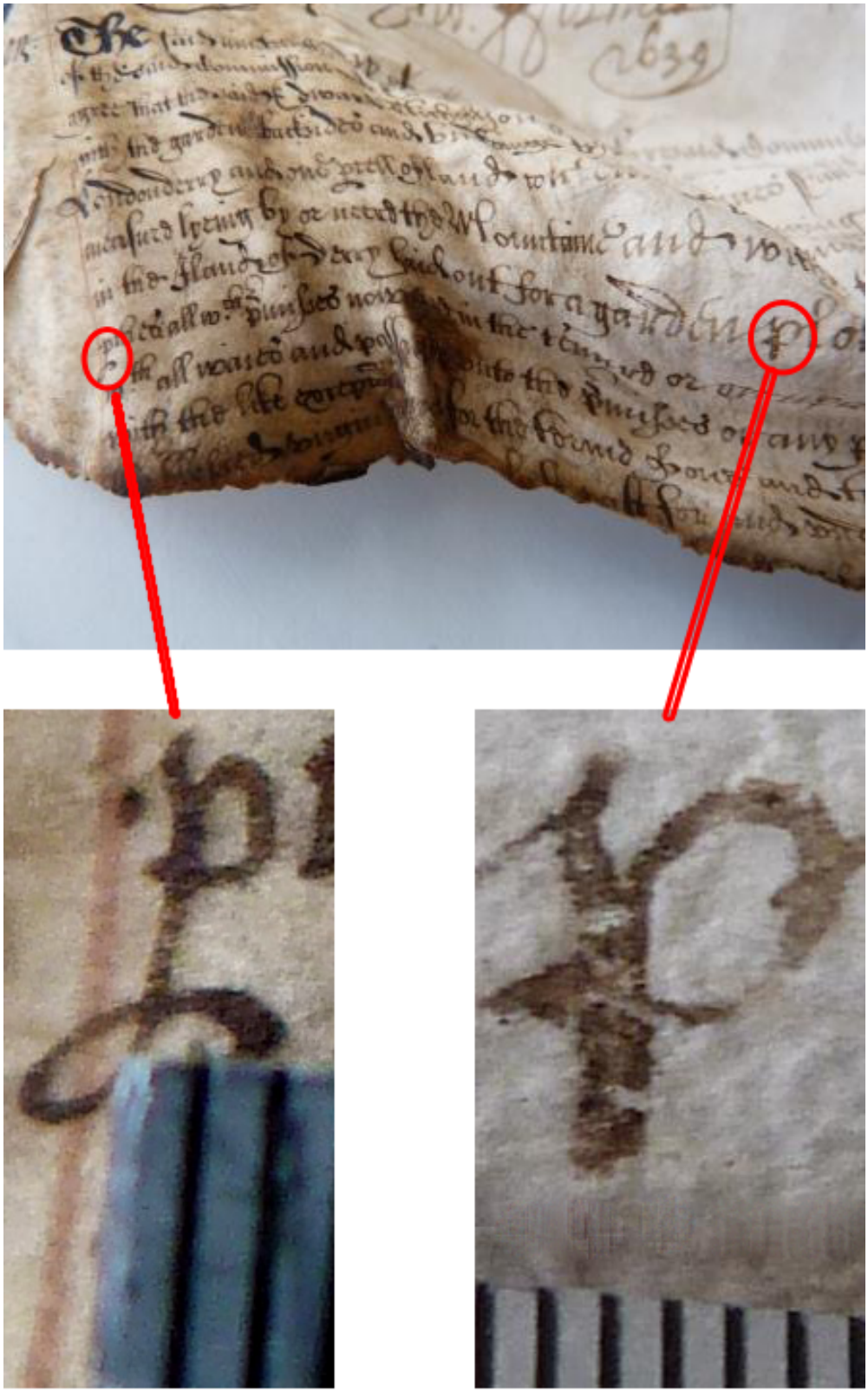

Figure 3.2: Top: A folio exhibiting variation in the amount of shrinking to different regions. Bottom: Close up views of letter p's from each region highlighting the different scales of letters that were once of the same size. (Reproduced with the permission of The Honourable The Irish Society and The City of London Corporation, London Metropolitan Archives)

Figure 3.3: An example of calcite formation, visible as the white powdery substance on the parchment surface. (Reproduced with the permission of The Honourable The Irish Society and The City of London Corporation, London Metropolitan Archives)

Gelatinisation. Gelatinisation is a degradation process in which the collagen fibres of the parchment change their structure irreversibly. The severity of the damage was rated either G1 (minor), G2 (full glass layer) or G3 (full penetration of parchment). All examples of gelatinasation were recorded at either G2 or G3, examples of which are shown in Figure 3.4.

Ink Loss. In some areas of the parchment, the ink has started to flake off (Figure 3.5 left), occasionally flaking off completely and leaving a lighter colour, presumably because the ink had protected the original parchment colour from dirt before flaking off (Figure 3.5 right).

Some conservation treatment has been previously applied to twenty one sheets of the book, however there is no documentation of when this was done or what techniques were used.

These sheets are far flatter than the untreated ones however the process

Figure 3.4: Examples of parchment gelatinisation. *Top*: G2 level, where the parchment is still somewhat pliable. *Bottom*: G3 level, where the parchment is fully gelatinised and rigid. (Reproduced with the permission of The Honourable The Irish Society and The City of London Corporation, London Metropolitan Archives)

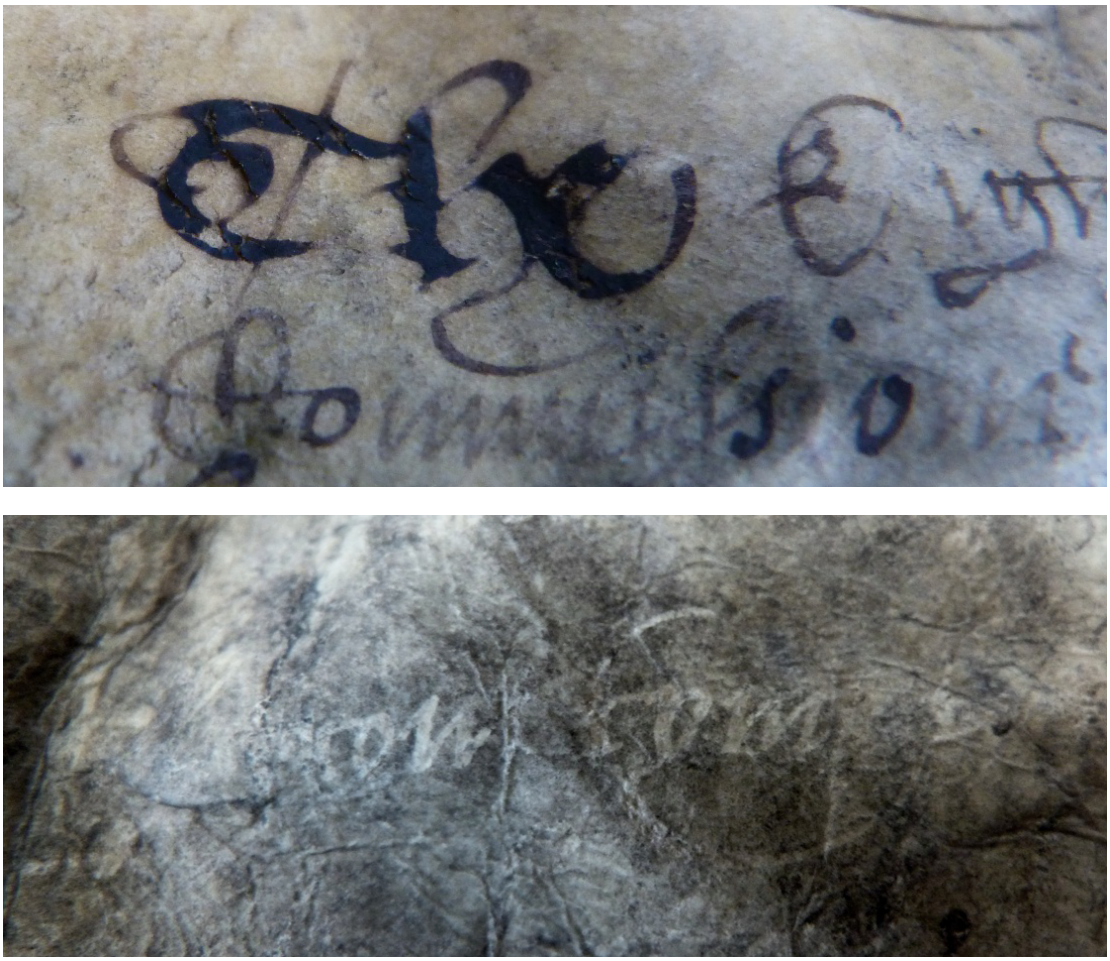

Figure 3.5: Examples of ink loss. Left: Small amounts of ink have started to flake off of the parchment. Right: The ink has completely flaked off, leaving the text somewhat visible as a lighter colour in contrast to the dirt-stained parchment around it. (Reproduced with the permission of The Honourable The Irish Society and The City of London Corporation, London Metropolitan Archives)

caused a considerable denaturing to the parchment. The sheets are far more gelatinised than the untreated ones, leaving them far more brittle. They have also become translucent meaning that ink on both sides of the folio is visible, which greatly hinders readability. This can be seen in Figure 3.6.

The conservation process for the remaining sheets needed to be far gentler to avoid damaging them in this way. The goal of this conservation was not to flatten the folios but rather to expose as much hidden text as possible for subsequent phases of the project.

The first step of the conservation is to clean the surface of the parchments using vulcanised rubber sponges (which are effective at removing soot and

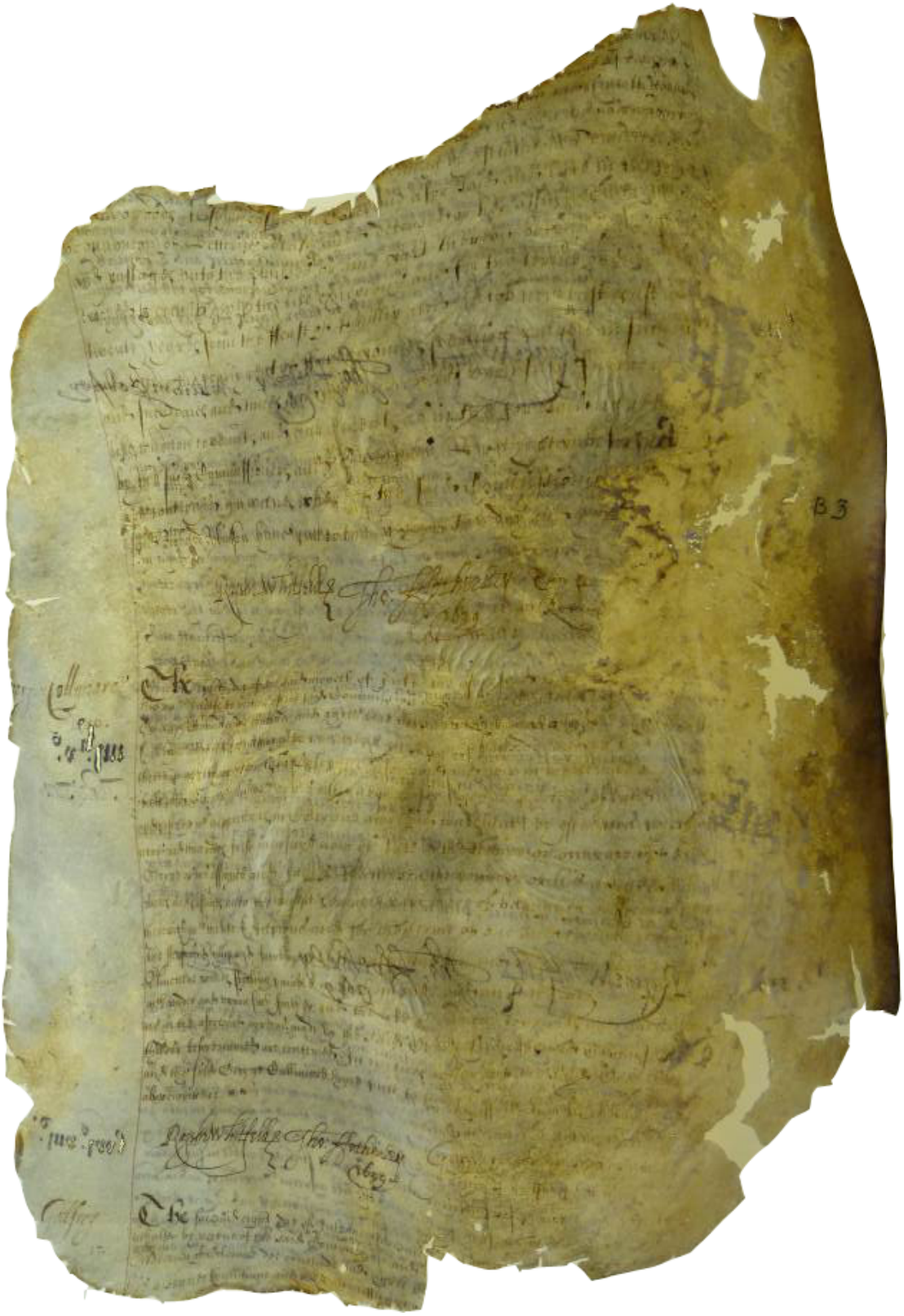

Figure 3.6: A previously treated folio. (Reproduced with the permission of The Honourable The Irish Society and The City of London Corporation, London Metropolitan Archives)

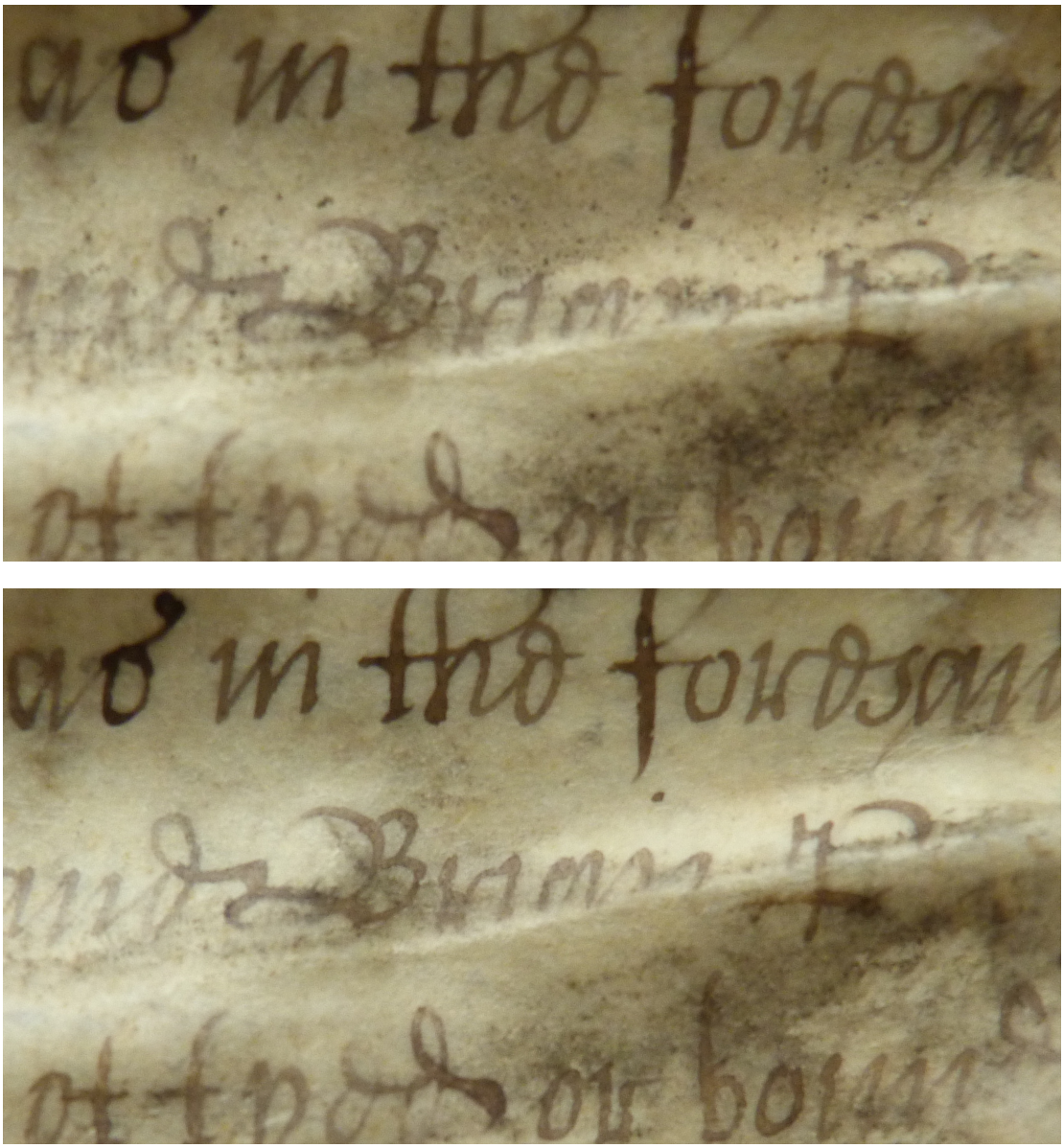

Figure 3.7: A region of text before (left) and after (right) gentle surface cleaning. (Reproduced with the permission of The Honourable The Irish Society and The City of London Corporation, London Metropolitan Archives)

smoke damage) and soft goat hair brushes. The goal of this is to remove as much surface dirt as possible so that it isn't later drawn deeper into the parchment structure. However, in areas where flaky ink is present, this step must be skipped to avoid further removing the ink. An example of the effect of the surface cleaning is shown in Figure 3.7.

The next step of the process is to humidify the parchment to make it less rigid and more pliable so that the creases can be opened.

The sheets were placed in a "*humidification sandwich*", consisting of Gore-

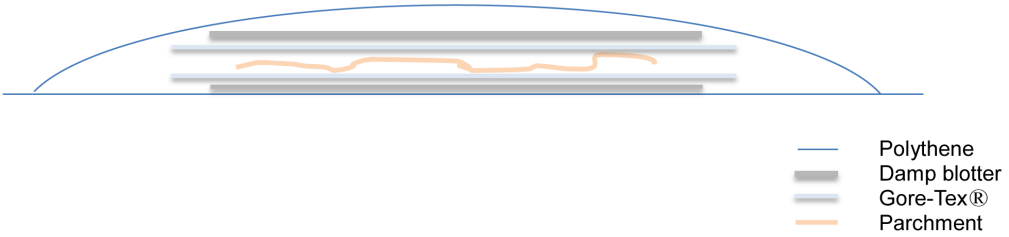

Figure 3.8: A diagram showing the arrangement used to humidify a parchment folio. (Reproduced with the permission of The Honourable The Irish Society and The City of London Corporation, London Metropolitan Archives)

Figure 3.9: A photograph of a folio being prepared for humidification. (Reproduced with the permission of The Honourable The Irish Society and The City of London Corporation, London Metropolitan Archives)

Tex and damp blotter. They were left in this state for approximately six hours. Figure 3.8 shows a diagram of this setup.

Areas of particularly shrunken parchment could be further softened using small "*humidity packs*" consisting of Laponite wrapped in thin non-woven polyester and flash spun high-density polyethylene fibres. These could be placed at desired locations in the sandwich to increase the humidity in those areas. Figure 3.9 a photograph of one of the sandwiches with humidity packs being applied.

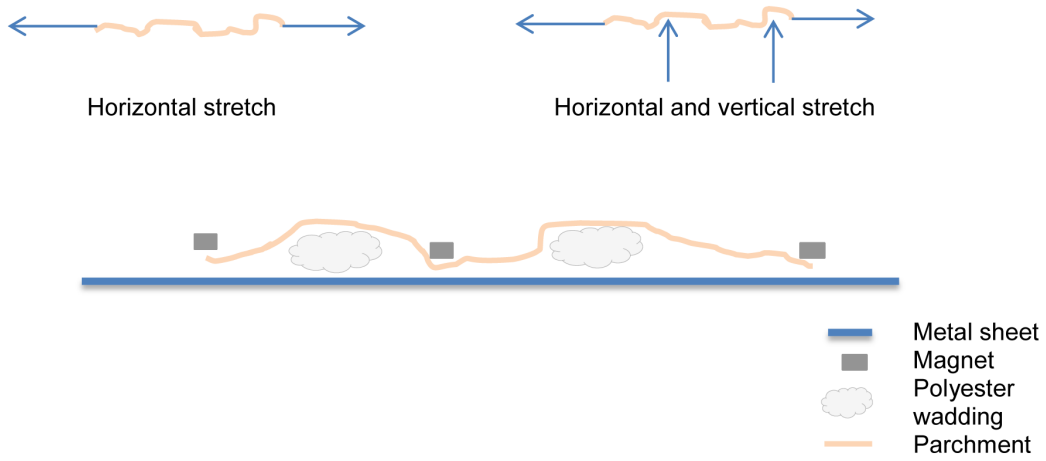

Figure 3.10: A diagram of the arrangement used to tension-dry a parchment folio following humidification. (Reproduced with the permission of The Honourable The Irish Society and The City of London Corporation, London Metropolitan Archives)

The final step of the process is to fix the parchments in a new shape and allow them to dry. The parchments were placed on metal sheets wrapped in thin blotter and covered with non-woven polyester, and then small magnets wrapped in felt were used to hold open creases while the parchment dries.

Certain creases in the parchments could not be pulled open horizontally due to surrounding distortions, and needed to be pulled open using a vertical force. To achieve this, polyester wadding was inserted underneath the creased area to pushing it open in a vertical direction, while being held in place horizontally with the magnets. A diagram and photograph of this configuration are shown in Figures 3.10 and 3.11.

Conservators were able to match up a number of small, stray fragments of parchment with larger in-tact sheets. These fragments were fixed back onto the larger sheets using gelatine coated tissue.

3.3 Discussion

The Great Parchment Book is extremely valuable to historians studying the colonisation of Northern Island, and has been described by the University of Ulster as a "*Domesday Book of Derry*" [London Metropolitan Archives, 2013]. Due to its its extremely damaged and fragile state, only a limited amount of

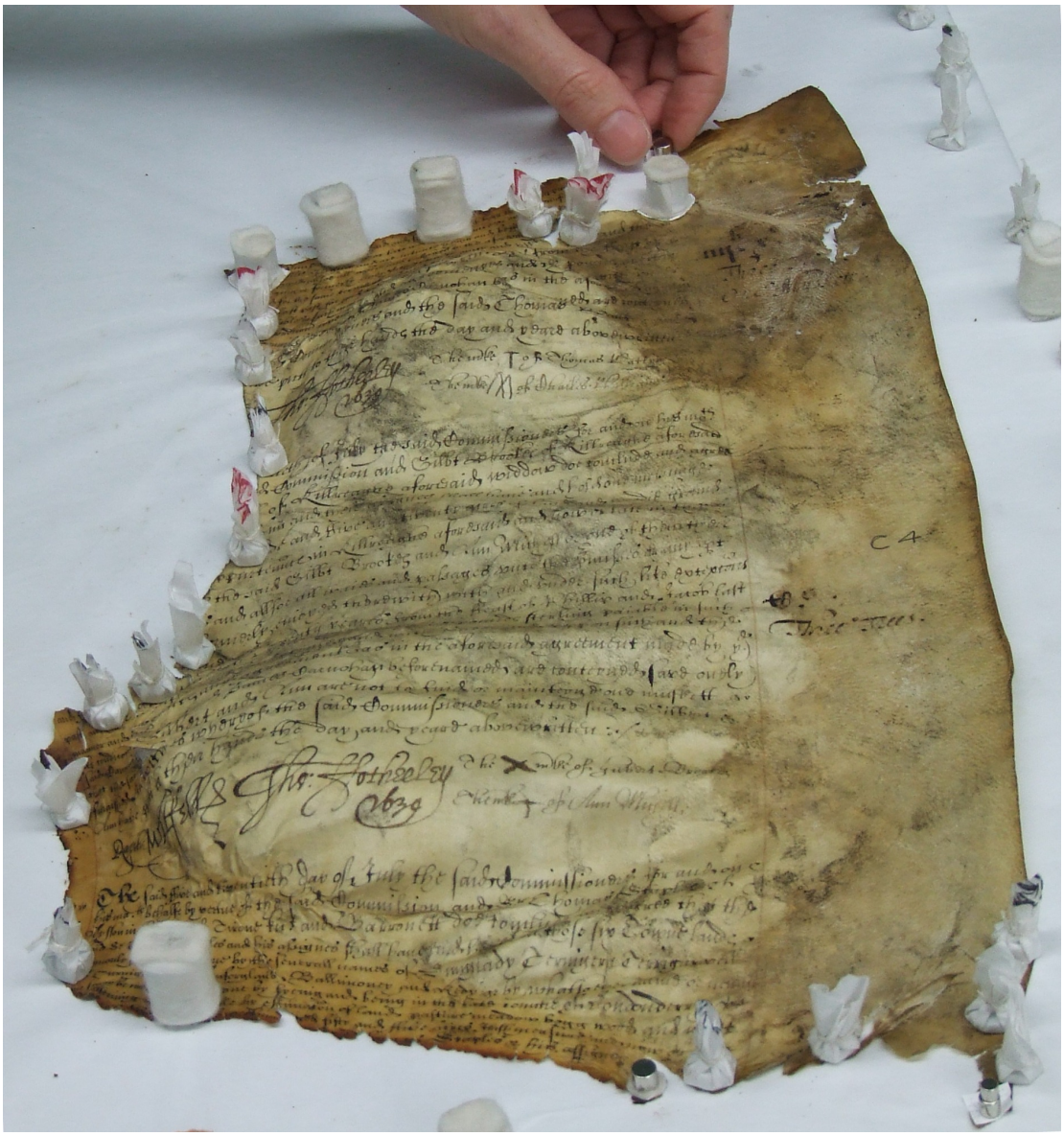

Figure 3.11: A photograph of a folio undergoing the tension-drying process. (Reproduced with the permission of The Honourable The Irish Society and The City of London Corporation, London Metropolitan Archives)

gentle conservation can be performed to improve its legibility. For this reason, a virtual restoration process was developed to increase the book's legibility and accessibility further than can be achieved with a conventional physical approach. In Chapter 4 we perform an analysis of the overall virtual restoration process and how it can be broken down into a pipeline consisting of several components, and in the subsequent chapters we discuss each component in detail.

Chapter 4

Problem Analysis

In Chapter 3, we introduced The Great Parchment Book and the joint project between London Metropolitan Archives and University College London. The goal of the project is to disseminate the content of the book in a readable way. This involves conserving, digitising, and transcribing each folio of the book. In this Chapter, we examine the elements of the project and discuss the problems associated with each.

4.1 Conservation

Parchment is a common medium for historical documents. As discussed in Section 2.1, it is made from limed animal hide and is extremely sensitive to changes in the environment. It can shrink, swell, wrinkle, and buckle if exposed to heat or humidity, introducing dramatic distortions to its shape [Woods, 1995]. The primary mode of restoring parchment damaged in this way is to flatten it to reveal text hidden by the creases [Woods, 1995, Clarkson, 1987, London Metropolitan Archives, 2013]; conservators at London Metropolitan Archives have told us that physically flattening the parchment in a responsible way methods is impossible using conventional conservation, since they are very delicate and brittle, and great care must be taken when handling them to minimize the risk of further damage. A thorough description of physical flattening methods and their limitations is given in Chapters 2 and 3.

To overcome these limitations of physical conservation, we propose a pipeline for digitisation and virtual restoration of the documents, which we discuss in subsequent sections.

4.2 Digitisation

When digitising a flat document, a single top-down image is sufficient to capture a high quality digitisation [Federal Agencies Digitization Initiative, 2010, Library of Congress, 2007]. However, as we have discussed, parchment deforms in such a way that an originally flat folio can adopt a very three dimensional shape when damaged [Woods, 1995]. In this case a single image would be insufficient to produce a high-quality digitisation since regions of the folio would be occluded by folds, and some visible regions of the folio would be imaged with foreshortening effects. For these reasons, we propose that each folio needs to reconstructed in 3D to produce a digitisation of sufficient quality.

In Chapter 2, we discussed many different methods of 3D reconstruction, each of which has its own strengths and limitations. In order to select an appropriate method, we must first define the requirements of our particular use case:

- **Fast**. Since a large number of documents need to be digitised it is important that the process is relatively fast.

- **Low cost**. We would like the acquisition process which we propose, or similar ones, to be usable by various different archives, libraries, and other cultural heritage institutions. Many such institutions have limited funding and so a method which relies on expensive, highly specialised equipment would not be ideal. Instead it should work with high quality but not overly expensive capture technology, ideally a technology which these institutions are already familiar with.

- **Non-damaging**. The folios of the Great Parchment Book (and other historical documents) are very fragile. It is therefore important that the imaging process does not damage them further through excessive handling or over-exposure to light.

- **Reconstruction Quality**. The acquisition process should generate geometry of high enough quality to capture the folds and creases of the document so that they can be flattened, and texture which allows the the text to be

clearly read. However, micro-geometry such as fibre structure or scratches on the document surface, and advanced reflectance models such as a complete BRDF do not need to be captured.

Another aspect of the digitisation process which needs to be considered is how to measure and document the quality of the digitisation. Specifically, when imaging flat documents for archival purposes, it is normal procedure for an archive to record the DPI (dots per inch) of the image, a measure of the frequency at which the image samples the document [Federal Agencies Digitization Initiative, 2010, Library of Congress, 2007]. It is important that such images meet standardised requirements for their DPI to ensure that the documents are imaged at sufficient resolution for their contents to be studied effectively. It would be useful, therefore, to derive a similar measure of sampling frequency for the 3D reconstructions which reports a quality measure in terms of DPI so that they can be in some way measured against the standardised guidelines. However, this is a non-trivial problem for 3D digitisations because they are typically generated from many different images taken from different viewing distances and viewing angles.

Based on these requirements, a detailed description of our chosen digitisation process, including a 3D reconstruction pipeline and a associated measure of sampling frequency, is given in Chapter 5.

4.3 Virtual Restoration

When performing a physical conservation process, flattening is almost always performed even if other treatments, such as surface cleaning or infilling of the text, is required since it makes the entire surface more accessible and also makes the folios easier to store [Woods, 1995, Clarkson, 1987, London Metropolitan Archives, 2013].

The same approach can be applied to the virtual restoration process. Since the primary form of damage to the parchment folios is in the form of geometric distortion, the main goal of the virtual conservation is to flatten the folios and help reveal the text. Current digital archival practices are designed to deal

with two-dimensional images [Federal Agencies Digitization Initiative, 2010, Library of Congress, 2007], so a flattened image of a folio would fit in more easily with these practices.

In Section 2.2, we discussed the observation made by Brown et al. [Brown et al., 2007] that the problem of document flattening is essentially the same as the problem of surface parameterization, a heavily studied field of computer graphics, which we discussed in Section 2.5. The objective of surface parametrization is to compute a map that flattens a 3D surface into the 2D plane while minimising the amount of distortion introduced. In Section 2.5 we introduced a number of surface parametrization algorithms which propose a variety of distortion metrics and methods to minimize them.

In this thesis, we are studying a related but more general problem where an existing surface (a damaged parchment folio), has been deformed by an unknown, arbitrary, and complex deformation which we would like to estimate and revert. This is a difficult problem because of the nature of the material. As we have already discussed in Chapter 2, parchment is an organic material and has only a semi-irregular internal structure of fibres, variations in thickness, and also contains additional irregularities such as hardened scar-tissue, meaning that it deforms in unpredictable and non-uniform ways when exposed to heat and other environmental changes. The damage to each folio is also often uneven, meaning that different regions of the folio will deform more than others. For example, the edges of folios in a book damaged in a fire will be affected more than the centres of the folios. In addition, various degrees of physical conservation, which are described in Chapter 3, have been applied to each folio, further changing their shape. It is the combination of these factors which make the deformation to the parchment very irregular, and causes the implicit assumption of standard energy-minimising surface parametrization algorithms to break down.

These algorithms assume that the surface being parametrized is the original, distortion-free surface and attempt to flatten it while minimizing the introduced distortion. The metrics of the distortion (such as measures of conformality [Lévy et al., 2002, Sheffer and de Sturler, 2001, Sheffer et al., 2005] or

isometry [Tenenbaum et al., 2000]) are computed from the 3D geometry of the surface. However, in our case, the geometry of a parchment folio already contains the distortion which we want to undo. In some sense, respecting these metrics would preserve the distortion while flattening the document, and the final 2D result would remain deformed.

The methods discussed in Section 2.2 are applied to books in a flatbed scanner [Wu and Agam, 2002], individual pieces of paper resting on uneven surfaces [Tian and Narasimhan, 2011], folded paper [Brown and Seales, 2000, Brown et al., 2007], and scrolls [Samko et al., 2014], all of which are isometric deformations. Using standard metric-preserving parameterization algorithms can be effective on these kinds of simple deformations, because the assumptions that the algorithms make about the deformations (eg. isometry) are appropriate. However, for more complex deformations such as those seen in damaged parchment, these assumptions no longer hold.

Figure 4.1 shows 3D reconstructions of two folios of The Great Parchment Book. In Chapter 5 we explain exactly how these these reconstructions are generated. However, we use them here to illustrate the problems we have just discussed. We globally flatten the two folios using Least-Squares Conformal Mapping [Lévy et al., 2002], which has previously been proposed for document flattening [Brown et al., 2007].

The folio in Figure 4.1 (Left) exhibits very severe wrinkling around its edges (seen in the bottom of Figure 4.1 (Left)) as well as a large crater-like deformation in its centre, giving it a high surface-area-to-perimeter ratio. Flattening this folio globally using LSCM results in drastic stretch deformations as shown in Figure 4.2.

The folio in Figure 4.1 (Right) is considerably less damaged than Figure 4.1 (Left). The result of applying LSCM is shown in Figure 4.3. While the result does not contain the same level of distortion as in Figure 4.2, the folio is still not properly un-distorted.

Figure 4.1: 3D reconstructions of two folios of The Great Parchment Book.

4.4 Proposed Approach

In light of the problems we have discussed, we propose a digital restoration pipeline, illustrated in Figure 4.4. The pipeline begins by capturing a 3D digital surrogate of a folio, as discussed in Chapter 5. The resulting 3D model can be used as the input for two different and complementary flattening approaches.

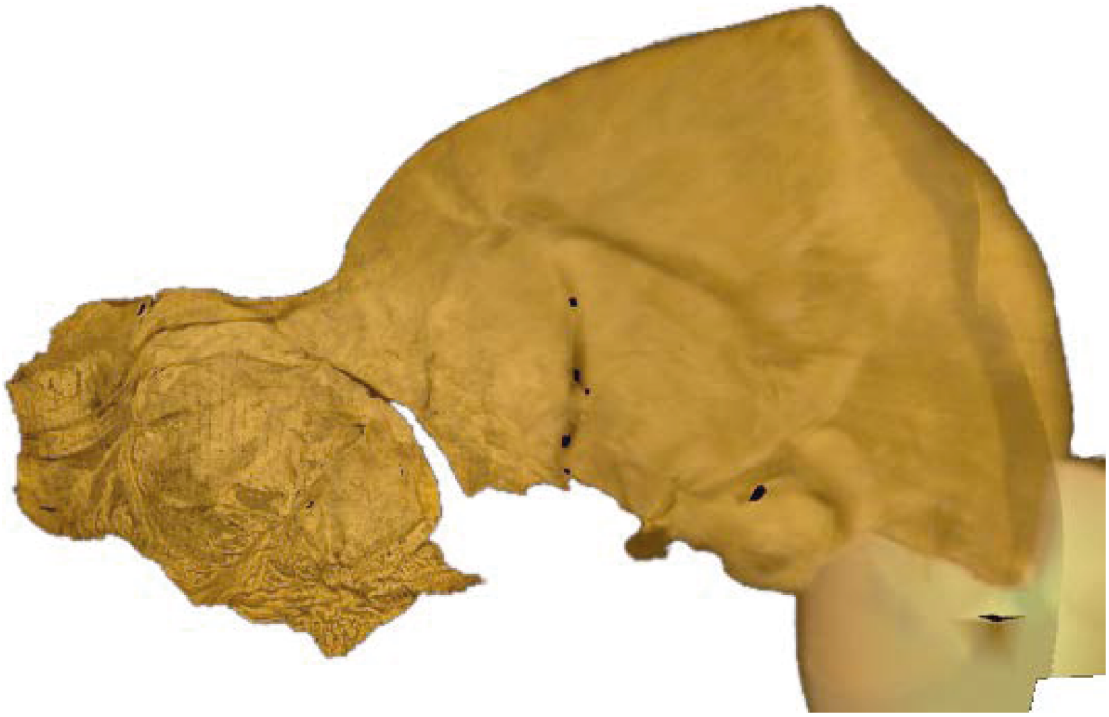

Figure 4.2: Global conformal flattening of the folio in Figure 4.1 (Left) causes large stretch distortions due to its large surface-area-to-perimeter ratio.

Local Flattening. The first approach is to provide the user with an interface which allows them to navigate over the three-dimensional surface of the document and flatten local subsets of the document on the fly.

This approach is analogous to the problem of map making. A global-flattening approach is equivalent to unwrapping the entire globe onto a single map of the earth which contains the well-known distortions typical of large map projections. Our local approach is equivalent to navigating over the 3D globe (in a Google Earth style fashion) and creating numerous local maps as we go, each of which contains far lower levels of distortion.

The goal of our viewer is that, as the user looks at a particular region of the document, it should be displayed in a way that is optimal in terms of its readability. This approach is fully explained in Chapter 6.

User-Guided Global Flattening. The above analogy to map-making breaks down in the sense that the earth was never once a flat object that was somehow deformed into a sphere. However, we know that the folios of the Great Parchment Book were once flat and some process has taken place which has deformed them into their current shape.

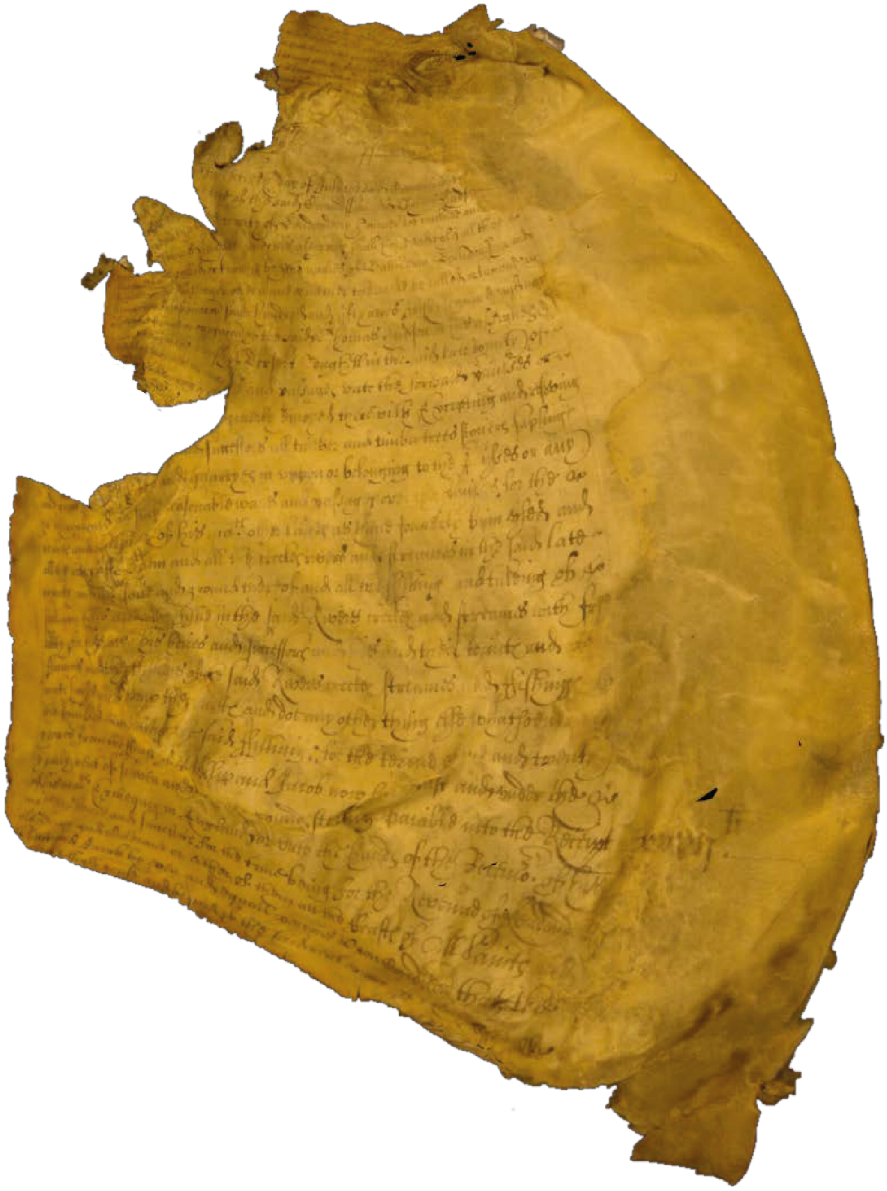

Figure 4.3: Global conformal flattening of the folio Figure 4.1 (Right).

Our second approach is to try to estimate the deformation that the original document underwent and revert it to virtually restore the document. While the parchments (such as the one shown in Figure 1) are severely deformed, we know that they originally were indeed flat and rectangular with a regular text structure. Such priors allow us to estimate the deformation of a parchment by analyzing its content. Inverting the estimated deformation flattens the geometry of the document, restoring it and its associated texture to its original state. This approach is fully explained in Chapter 7.

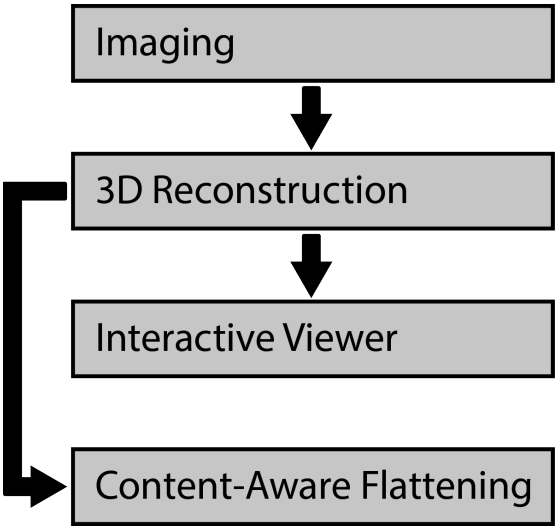

Figure 4.4: Our proposed virtual restoration pipeline.

Chapter 5

Imaging and Reconstruction

In this chapter we discuss the first step of our virtual restoration pipeline, namely the 3D acquisition process. We first discuss our choice of method, and then detail the imaging process and subsequent algorithms used to generate 3D models of the parchment folios. Finally we also introduce a method for assessing the quality of the 3D reconstructions in terms of *dots per inch* (DPI) since this is the standard way to report quality when digitising archival material.

5.1 Choice of Method

The method which was used was motivated by a number of factors which were laid out in Chapter 4. Possible 3D acquisition approaches include:

Laser Scanning. This approach provides very accurate reconstruction of geometry and has no problems when reconstructing objects with low surface texture. However, it requires expensive, specialised equipment which most archives and libraries would not have access to. Also, if the scanning subject contains self-occluding geometry, the scanning process becomes very time-consuming since number of different scans are required to capture the entire surface. A possible approach to make this process of taking multiple scans easier would be to mount the scanner on a robotic arm and use an algorithm to select optimal viewing angles. This would, however, increase the cost and the requirement of specialist equipment. We discuss this idea further in Chapter 8.

Structured-Light Scanners. These also generate very high accuracy reconstructions, and can scan low texture objects since the light pattern effectively serves as texture. The approach, however, suffers from a similar set of disadvantages

to conventional laser scanning, namely the need for specialist equipment, and the problem of self-occlusions necessitating multiple time-consuming scans.

Structure-from-motion. This method, unlike laser and structured-light scanning, is a passive approach and doesn't require specialist equipment to produce light-patterns. The only necessary hardware for this approach is a camera of sufficient resolution, and adequate lighting equipment to evenly illuminate the parchment. It should be noted that high-end SLR cameras can be very expensive, even more so than some commercial scanners. However, archives and libraries often have dedicated imaging facilities which will almost certainly already possess high-quality digital cameras. The method also suffers from self-occlusion problems, so many images are required to capture the entire surface, however taking multiple images with a camera is a much faster process than taking multiple laser scans. The structure-from-motion approach can fail when trying to reconstruct low-texture objects, however this is not a problem for our use-case since the parchment folios contain a sufficient amount of texture, especially in the most important regions where there is text.

For these reasons we opted for the structure-from-motion approach for the 3D acquisition process. In the remained of this chapter, we discuss the specifics of our approach.

5.2 Data Acquisition

The first step of the acquisition process is to capture a set of overlapping 2D images which covers the entirety of the parchment. The parchments were imaged using a hand-held DSLR camera (Canon 5D Mark III). Each parchment was placed on a table covered with a black velvet cloth (to provide a matt background) and illuminated with three large, evenly spaced diffuse lights to provide uniform illumination and minimise the amount of shadow cast on the parchment due to self-shadowing. We surround the parchments with a ring of ball-bearings (some shiny and some diffuse) which could later be used to reconstruct the lighting conditions accurately [Debevec, 1998][1], and a

[1]It did not prove necessary for us to do this in the course of this project. However, we would still recommend this practice to others performing similar imaging projects since it significantly increases the richness of the resulting data set in exchange for very little expense.

ColorChecker calibration target is placed on the table next to the parchment, providing a measure of scale and colour calibration.

For each folio, we first take a set of images (typically between eight and ten) in a circular formation such that the entire parchment and all the ball-bearings are visible in each image. We then take many more close-up images, making sure to cover the entire surface of the folio thoroughly. For highly distorted areas of the parchment where the text has shrunk to a very small size, we use a macro lens to obtain extreme close-up images of the text. Typically, a single image set will contain between fifty and sixty images, but can sometimes be as large as eighty or ninety images for extremely deformed folios, or as low as twenty or thirty for relatively flat folios.

Algorithms exist for automatically selecting optimal camera viewpoints [Ahmadabadian et al., 2014]. We will discuss the possibility of introducing such methods in Chapter 8, however as our process was carried out, the selection of camera views is dependent on the judgement and experience of the human operator. It is their responsibility to ensure that the image set adequately covers the entire folio and is free from images containing unacceptable levels of blur from either depth-of-field effects or motion blur.

Figures 5.1 and 5.2 show the imaging set-up and an example a complete image set for a single folio.

5.3 3D Reconstruction

In this section we describe our 3D reconstruction pipeline. When an image set has been captured for a folio, it can be used to generate a point cloud reconstruction of the geometry folio while also computing a camera calibration for each of the images, which can be used to compute a closed surface represented as a triangle mesh. The camera calibrations can then be used to map the colour information from the images back onto the reconstructed geometry to generate a fully textured reconstruction.

Figure 5.1: A photograph of the imaging process.

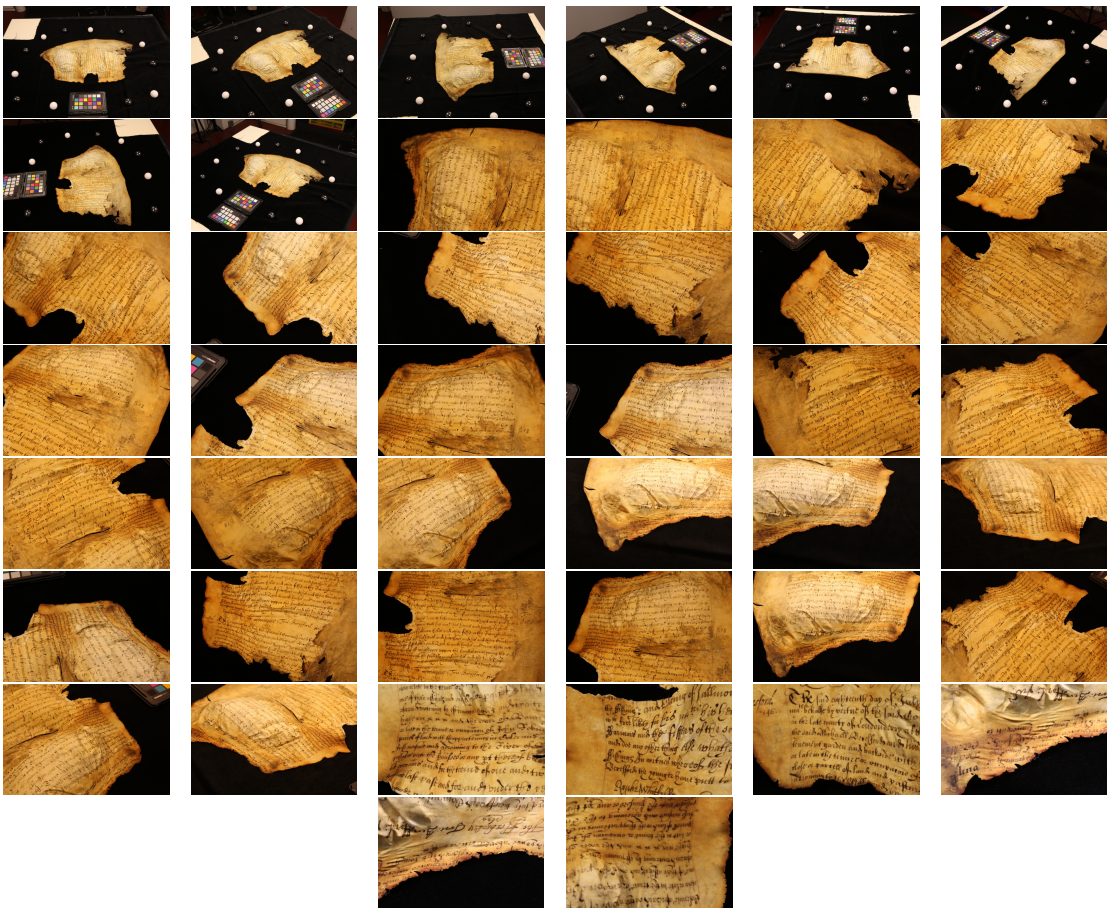

Figure 5.2: An image set for a single parchment folio.

5.3.1 Structure-from-Motion and Surface Reconstruction

We process the image sets with Wu et al.'s VisualSFM [Wu, 2011] software which uses Wu et al.'s GPU implementation of SIFT [Lowe, 2004, Wu, 2007] and their multi-core bundle adjustment algorithm [Wu et al., 2011] to generate a sparse 3D reconstruction using structure from motion. We then apply Furukawa and Ponce's PMVS algorithm [Furukawa and Ponce, 2010] to generate a dense point reconstruction, examples of which are shown in Figure 5.10(*Top*).

The PMVS algorithm is a widely used dense multi-view-stereo reconstruction algorithm, and both it and VisualSFM's structure from motion procedure perform very well on even highly unstructured image sets containing variations in lighting, image exposure, lens type, etc. Both VisualSFM and PMVS are open-source, support multiple operating systems, and are designed to be used together, making them an ideal choice for an end-user-friendly digitisation pipeline. They are discussed in more detail in Section 2.4.

This process also computes, along with the point reconstruction, calibration parameters for each input image. These parameters are a focal length f, a 3×3 camera rotation matrix $\mathbf{R}$, and a 3-vector camera translation $\mathbf{t}$.

From these parameters, we can determine the camera viewing direction $\mathbf{v}^{\mathcal{C}}_{\text{view}}$ as:

$$\mathbf{v}^{\mathcal{C}}_{\text{view}} = \mathbf{R}^T \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix}, \tag{5.1}$$

and the position of the camera's optical centre $\mathcal{O}^{\mathcal{C}}$ as:

$$\mathcal{O}^{\mathcal{C}} = -\mathbf{R}^T \mathbf{t}, \tag{5.2}$$

The reconstruction is computed up to an arbitrary scale, so the distances in the resulting object space do not correspond to the true distances in real-life space. To correct for this we allow a user to mark points on the colour checker which are a known distance apart in reality and then triangulate their positions in object space to compute a scaling factor which can be used to scale distances in object space to match those in real-life space.

The next step in our pipeline is to compute a triangle mesh from the dense point cloud, for which we use Kazhdan et al.'s Poisson Surface Reconstruction algorithm [Kazhdan et al., 2006]. This algorithm requires very little parameter-tuning (we use the exact same parameters for every reconstruction), is resilient to noisy data, and is able to interpolate holes in the point reconstruction very well. The algorithm makes use of the normal vectors associated with each point that are generated as part of the PMVS output, making it a natural choice to follow PMVS. We use the Poisson Surface Reconstruction implementation provided in MeshLab [Visual Computing Lab of CNR-ISTI, 2005]. Examples of reconstructed meshes are shown in Figure 5.10 (*Middle*).

As can be seen in Figure 5.10(*Top*), the point clouds contain holes in certain areas. The meshing process will smoothly interpolate a surface over the hole. If the hole was caused by poor coverage of that region in the image set, this could cause ghosting artefacts in the texture since the interpolated mesh surface may not be accurate, resulting in blurry text in the reconstruction. However, if the hole was caused by the absence of text in the region (and hence a lack of texture features for the reconstruction algorithm to detect), this is less of a problem since we care most about those regions of the folio containing text. In Chapter 8, we will discuss possible future work to help avoid the presence of such holes in the reconstruction.

5.3.2 Texture Mapping

The final part of our reconstruction pipeline is to generate texture maps for the triangle meshes. We use the same texture-atlas generation method as Esteban and Schmitt [Esteban and Schmitt, 2004], originally proposed by Schmitt and Yemez [Schmitt and Yemez, 1999], since it is simple to implement and avoids having to compute a texture parameterization for the mesh.

Each triangle in the mesh is mapped to its own right-angled triangle of edge length N in the texture atlas. For each vertex $\mathbf{p}$ in the mesh, and for each input camera $\mathcal{C}_i$, we assign a weight $\tilde{w}_{\mathbf{p}}^{\mathcal{C}_i}$ according to how well $\mathcal{C}_i$ sees $\mathbf{p}$. The weighting scheme is adapted from Buehler et al.'s Unstructured Lumigraph interpolation [Buehler et al., 2001] scheme, and takes into account visibility,

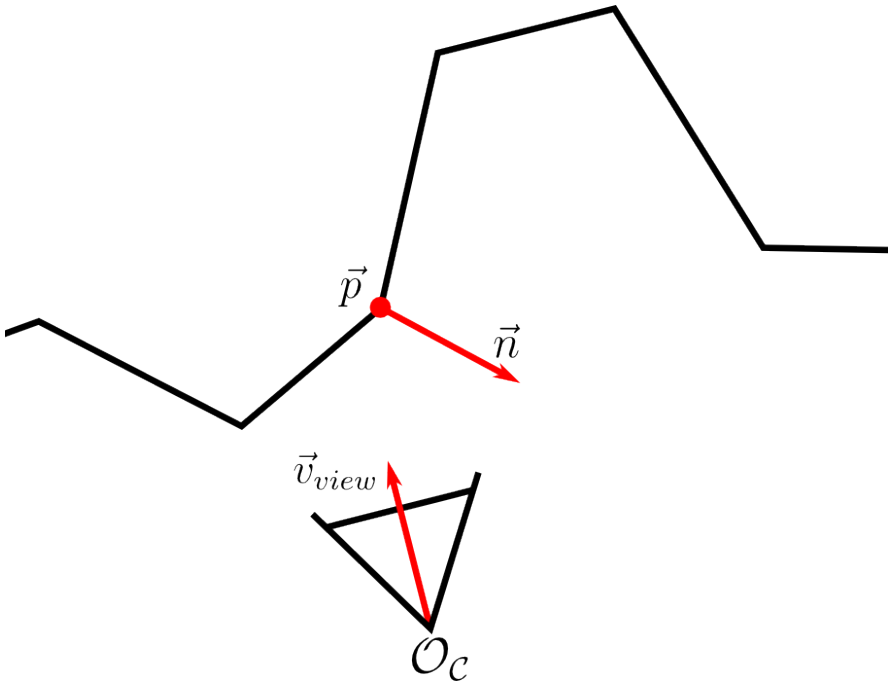

Figure 5.3: Illustration of a camera $\mathcal{C}_i$ with optical center $\mathcal{O}_{\mathcal{C}_i}$ and viewing ray $\mathbf{v}_{view}$, and a point $\mathbf{p}$ on a mesh with normal $\mathbf{n}$. We define a weighting scheme to determine the appropriateness of using $\mathcal{C}_i$ to texture the surface around $\mathbf{p}$

relative angle, and resolution of the vertex in each camera.

For a vertex $\mathbf{p}$ with normal vector $\mathbf{n}$, and a camera $\mathcal{C}_i$ in the set of input cameras C, with center of projection $\mathcal{O}_{\mathcal{C}_i}$ and viewing ray $\mathbf{v}_{\text{view}}$, as illustrated in Figure 5.3, we define the penalty terms as:

$$
\begin{aligned}
E_{\text{ang}} &= \arccos(\mathbf{n} \cdot \mathbf{v}_{\text{view}}), \\
E_{\text{res}} &= \|\mathbf{p} - \mathcal{O}_{\mathcal{C}_i}\|, \\
E_{\text{vis}} &= \begin{cases} 0, & \text{if } \mathbf{p} \text{ is visible in } \mathcal{C}_i \\ \infty, & \text{otherwise} \end{cases} \\
E_{\text{total}} &= w_{\text{ang}} E_{\text{ang}} + w_{\text{res}} E_{\text{res}} + E_{\text{vis}},
\end{aligned}
$$

and compute a weight $\tilde{w}_{\mathbf{p}}^{\mathcal{C}_i}$ as described in Section 2.4.7:

In our system we choose the weights $w_{\text{ang}} = 5$ and $w_{\text{res}} = 1$ since our experiments with various values show these to work well. The visibility term, E_{vis}, is computed by first checking that the projection of $\mathbf{p}$ into $\mathcal{C}_i$ lies within the image, and then testing for occlusions by performing a ray cast from $\mathbf{o}_i$ to $\mathbf{p}$ and

checking that the ray does not intersect the mesh in before reaching $\mathbf{p}$.

The angular term, E_{ang}, prefers cameras whose viewing angle is as close to parallel with the surface normal at $\mathbf{p}$. This aims to reduce blurriness in the texture by avoiding the use of images which contain foreshortening effects.

The resolution term, E_{res}, prefers cameras whose optical centres are close to $\mathbf{p}$. These close-up images capture the surface details in the highest resolution and are therefore ideal for generating the highest resolution textures possible. Note that other considerations related to image resolution are the presence of motion blur and depth-of-field effects in the images. These could cause a texture feature on parchment to appear blurry even if it is close to the camera's optical centre, making it unsuitable for use in computing the texture. Our method, however, does not take these into account and instead relies on care being taken in the imaging process to avoid these effects being present in the image set. However, our process could be made more robust by automatically detecting these effects and incorporating them into the penalty terms.

For a given vertex, we rank each input camera according to their total penalty and then define a threshold, T_{adaptive}, as the 3^{rd} largest penalty value. The weight $\tilde{w}_{\mathbf{p}}^{C_i}$ is then computed as:

$$w_{\mathbf{p}}^{C_i} = \max\left(1 - \frac{E_{\text{total}}}{T_{\text{adaptive}}}, 0\right),$$
$$\tilde{w}_{\mathbf{p}}^{C_i} = \frac{w}{\sum_{C_j \in \mathcal{C}} w_{\mathbf{p}}^{C_j}},$$

corresponding to Unstructured Lumigraph interpolation (discussed in Section 2.4.7) [Buehler et al., 2001] with $k = 3$, ensuring that only the two cameras with lowest penalty are given non-zero weights, while continuous variations in the penalties lead to smooth variations of camera weights.

To generate the texture for triangle T with vertices $\mathbf{p}_0$, $\mathbf{p}_1$, and $\mathbf{p}_2$ (each of which has been assigned the two cameras that see it best), we form the set of cameras $\mathcal{C}_T$ containing the cameras associated with the vertices of T. $\mathcal{C}_T$ will contain at most six cameras but will often contain fewer than six since it is likely that neighbouring vertices will be best seen by the same cameras. Then for

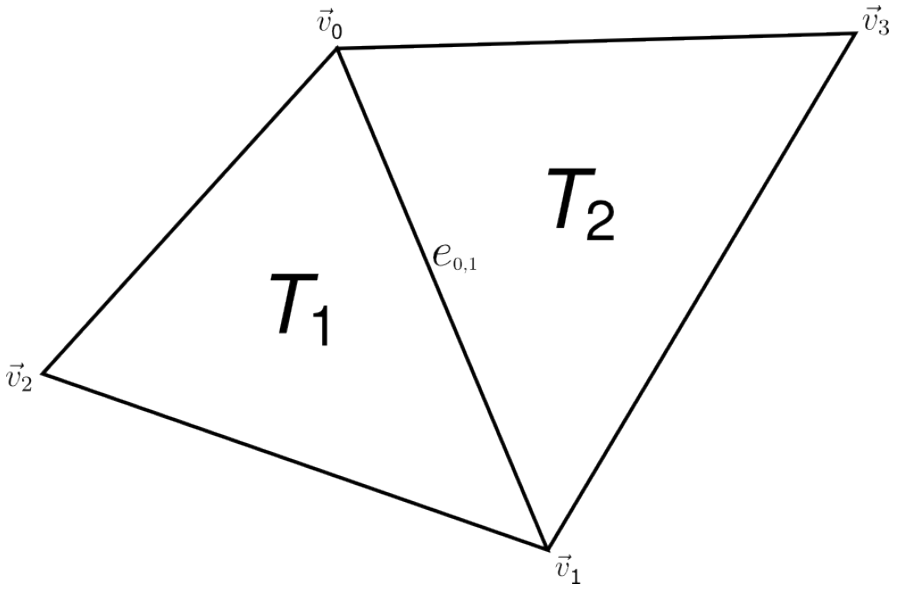

Figure 5.4: When texturing two adjacent triangles T_1 and T_2, the cameras selected for vertices $\mathbf{p}_0$ and $\mathbf{p}_1$ are independent of which triangle we are considering. Therefore the texture will blend smoothly over the edge $e_{0,1}$

each camera $C_i \in \mathcal{C}_T$, we set the alpha value at each vertex $\mathbf{p}_j$ of T to be $\tilde{w}_{\mathbf{p}_j}^{C_i}$, i.e., the weight of C_i at the j^{th} vertex of T, and then render the triangle into the appropriate section of the texture map.

One of the most important considerations of our camera assignment scheme is that the blending between cameras should be smooth and the texture should not contain any sudden discontinuities. Consider generating the textures for two neighbouring triangles T_1 and T_2 which share two vertices $\mathbf{p}_0$ and $\mathbf{p}_1$ and an edge $e_{0,1}$, as illustrated in Figure 5.4. Since the best cameras for $\mathbf{p}_0$ and $\mathbf{p}_1$ are chosen independent of which triangle we are considering, the texture in T_1 will blend smoothly over $e_{0,1}$ into the texture in T_2.

If we just consider a single triangle T, however, there may still be discontinuities in its texture if one of the cameras used to texture it cannot see all of T. This can only happen at a visibility boundary, where a camera sees some but not all of the vertices of a triangle. Consider the scenario in Figure 5.5, which shows a visibility mask for a camera $\mathcal{C}$ over some triangle mesh. The red triangles contain no vertices visible in $\mathcal{C}$, the green triangles have all vertices visible in $\mathcal{C}$, and the yellow triangles (such as the triangle labeled T) have some but not all vertices visible in $\mathcal{C}$. If $\mathcal{C}$ is used to texture triangle T, the resulting texture

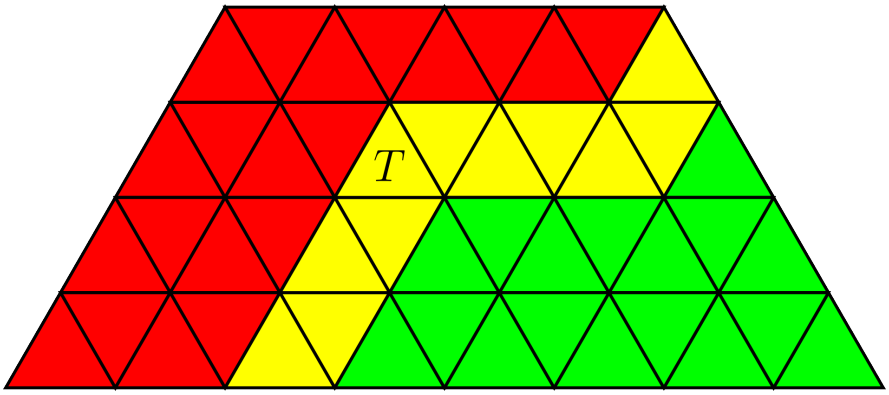

Figure 5.5: An example visibility mask of camera and triangle mesh. Green triangles are completely visibly by the camera, red triangles completely out of view, and yellow triangles partially visible.

could contain some erroneous artefact since $\mathcal{C}$ may have some blending weight in parts of the triangle that it cannot actually see. To avoid this situation, we erode the visibility mask of $\mathcal{C}$, as illustrated in Figure 5.6, so that a vertex is only considered visible by $\mathcal{C}$ if each triangle incident on that vertex is completely visible in $\mathcal{C}$. We use this eroded visibility mask in our computation of E_{vis}.

5.4 Assessing Reconstruction Quality

Professional archival standards for document digitisation require minimum resolutions for the resulting raster images, and many standardised benchmarks of archival quality images have been defined depending on the type of document being imaged [Federal Agencies Digitization Initiative, 2010, Library of Congress, 2007]. In the case of planar 2D artefacts that are imaged with flatbed scanners or in a fronto-parallel camera image, this minimum resolution is expressed in dots per inch (DPI). That is, a measure of the sampling frequency of the document being imaged. Specifically, it is the number of samples (i.e. image pixels) that are taken in the space of one linear inch on the surface of the document. In the case of the Great Parchment Book, the folios should ideally be imaged at 600 DPI, and at a minimum of 300 DPI.

Measuring DPI is simple when imaging a single flat object from a fronto-facing viewpoint. In our case, however, with a 3D reconstruction texture generated by blending many different images from different viewing distances and viewing angles, the effective sampling density of the reconstructed parchment

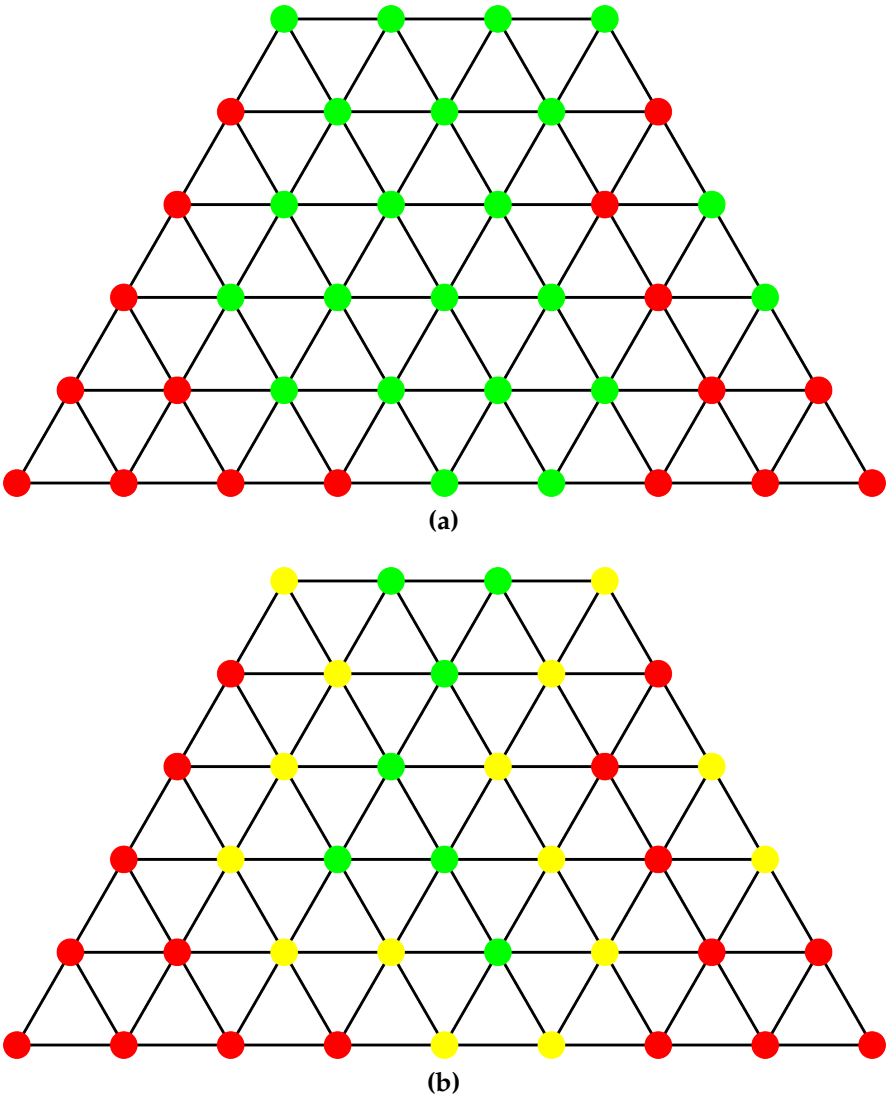

Figure 5.6: Visibility erosion. Left: Green vertices are visible in the camera and red vertices are not. Right: The erosion step removes the yellow vertices from the visible set. Only the green vertices are considered visible for the purposes of texturing.

varies across the parchment surface, dependent on the imaging conditions of the images contributing to each surface point. Therefore, assigning a single DPI quality label would not sufficiently characterise the dataset. Also, since we cannot guarantee that every point on the manuscript surface is imaged from a fronto-planar viewpoint, there will inevitably be a degree of anisotropy (or stretch) in the sampling.

When rendering the texture atlas for a parchment folio, we could fix the resolution of the atlas to be arbitrarily large without increasing the amount of detail present in the rendering. Therefore, we cannot use the resolution of

the texture as an indicator of the DPI. Instead we want to assess the *effective DPI* of the texture. That is, a measure of the frequency at which details on the parchment surface are sampled by the imaging and reconstruction process. As previously explained, the different viewing angles of the image set cause a certain amount of anisotropy in the sampling. This means that the effective DPI will be greater in one direction on the parchment surface and smaller in the perpendicular direction.

In order to allow for an assessment of the reconstruction quality in familiar terms, we propose looking at the *distributions* (or histograms) of both effective DPI and local stretch in the sampling across the parchment. Figure 5.7 shows examples of these distributions for a single parchment. The histogram of effective DPI shows that the majority of the mesh vertices are sampled at over 600 DPI. It also shows that the distribution is bi-modal, with a small second cluster of vertices sampled at around 200 DPI. Along with these histograms, heatmap renderings (shown in Figure 5.8) can also be generated to show exactly where on the manuscript surface these variations in sampling occur. From these, we see that these low-DPI vertices are mostly on the edges of the folio, which were most likely imaged less thoroughly due to the absence of text or other important features. We argue that these histograms and heatmaps provide an excellent way to gauge the quality of a dataset in terms easily communicated to archivists.

We determine the local, effective sampling density at each mesh vertex $\mathbf{p}$ by looking at the mapping of the camera image that contributed the most to the texture at $\mathbf{p}$. Let the corresponding camera's projection matrix be P, we approximate its perspective mapping from the image to the mesh domain as being locally affine, which allows us to express this mapping around $\mathbf{p}$ by its Jacobian $J_P(\mathbf{p})$ [Heckbert, 1989]. Its determinant $\det J_P(\mathbf{p})$ denotes the inverse surface area an image pixel projects to. Hence:

$$d = \sqrt{\det J_P(\mathbf{p})}$$

is the local mean sampling density in DPI (assuming units of inches). However, for grazing camera views, the sampling may be anisotropic. The level of

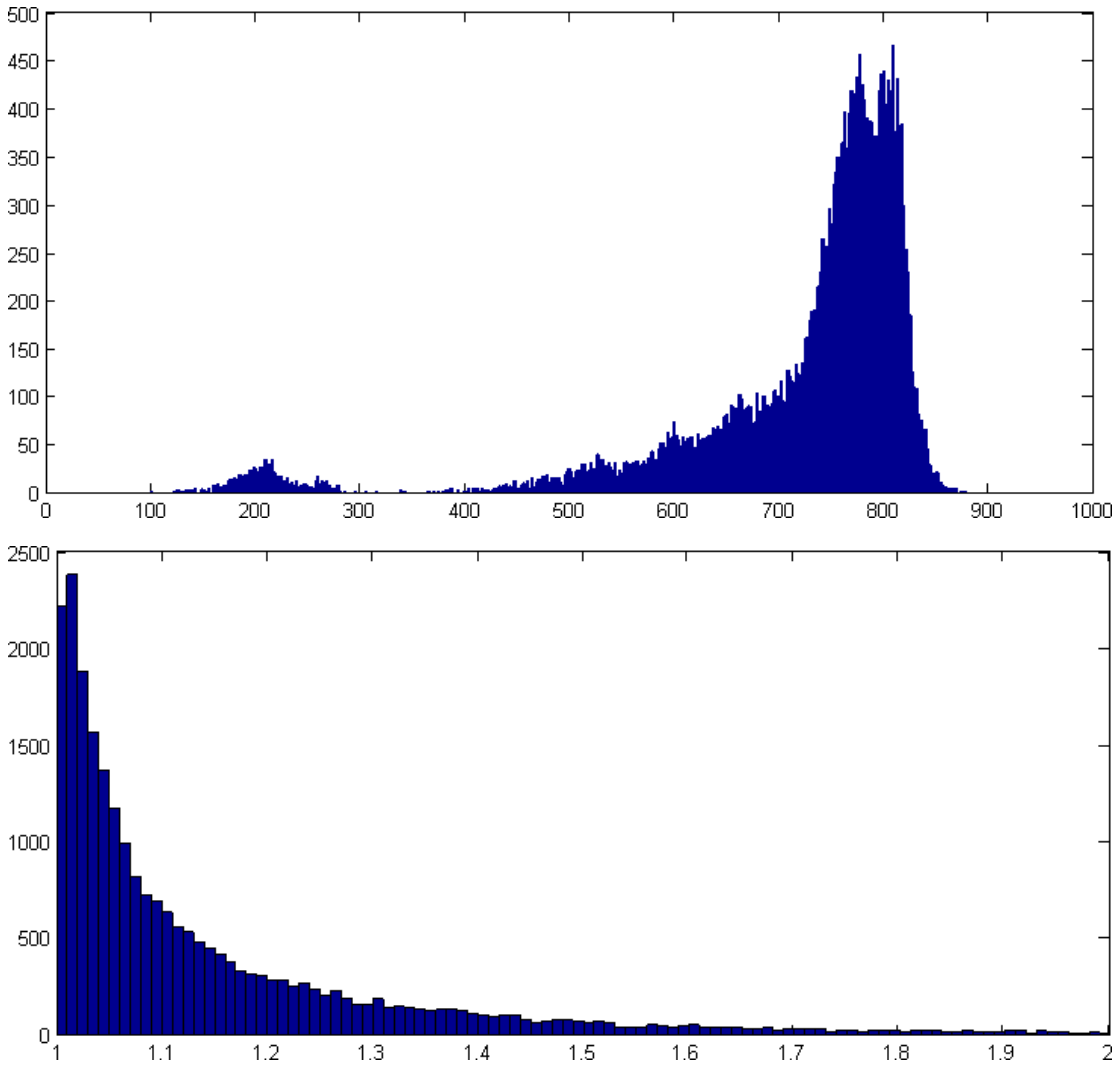

Figure 5.7: *Top:* Histogram of effective DPI. Most mesh vertices are sampled at over 600 DPI, with a much smaller number of vertices having been captured at approximately 200 DPI. *Bottom:* Histogram of local sampling stretch. The vast majority vertices exhibit significantly less than 1:1.3 anisotropy.

anisotropy is given by the Jacobian's condition number (the ratio of its eigenvalues):

$$a = \kappa(J_P(\mathbf{p}))$$

implying that the local sampling is $d\sqrt{a}$ DPI in the direction of maximum stretch and $d/\sqrt{a}$ DPI perpendicular to it.

5.5 Results

Table 5.1 shows various statistics about the Great Parchment Book imaging process. From these statistics we can see the scope of the imaging process,

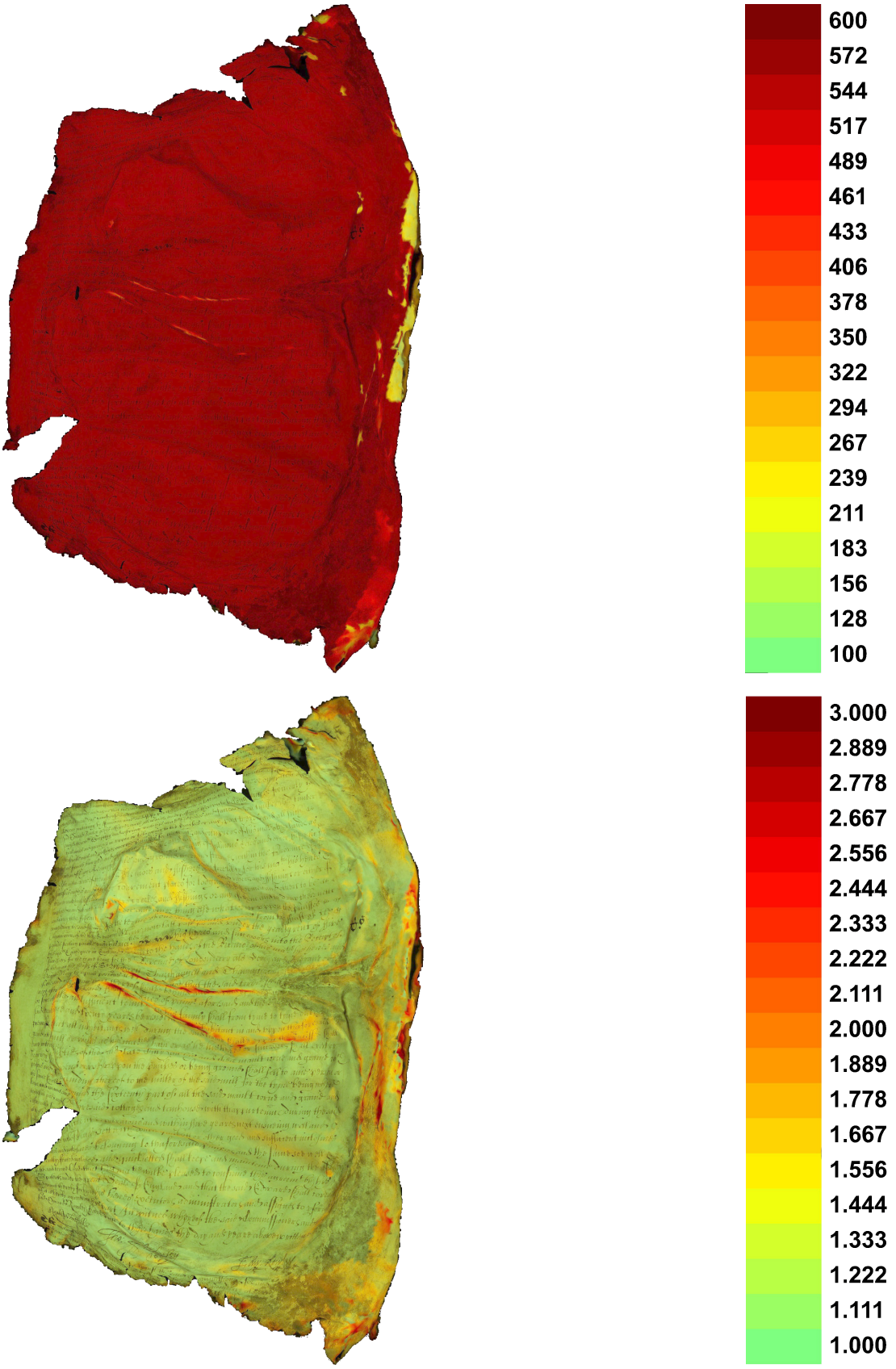

Figure 5.8: *Top:* heatmap of sampling density. Higher values (red) indicates the parchment was imaged at a high resolution. *Left:* heatmap of sampling anisotropy. Low values (green) are preferred and indicated that the parchment was imaged with a front-facing camera view.

Overview	Total folio sides	305
	Total images	14941
	Folios requiring macro images	260
Images per side	Mean	49
	Max	89
	Min	11
	Standard deviation	10.2
Time taken	Time per side	10-15 minutes
	Sides per day	12.7
	Total days	24

Table 5.1: Statistics demonstrating the scope of the Great Parchment Book imaging process.

with nearly 15,000 images taken to cover 305 folio sides. Note that the total number of sides imaged (305) is less than double the total number of folios (165) because a number of folios have one side blank. It also shows the variation in the different image sets, which reflects the variation in shape of the different folios. As previously mentioned, a typical image set for a single side contains between 40 and 60 images, but depending on the complexity of a folio's shape this number varies from as low as 11 images to as high as 89.

The time investment required for such a project is also apparent, with the imaging sessions spread out over 24 days. These were not 24 consecutive days, but were rather organised according to the availability of facilities at LMA. Typically one or two imaging sessions would be performed in a given week, each lasting half a working day. This arrangement of half days with days off in between meant that the entire process was spread out over months, but a significant benefit is that it kept the physical strain of capturing the images to a manageable level.

Figure 5.9 shows an example of the camera positions used to image a single side of a folio. A small number of wide-baseline images are taken in a circular arrangement. This arrangement is used for every folio with very little variation. The bulk of the images, however, are taken closer to the folio and their arrangement is chosen according to the specific shape of that folio. Notice, for example, the arrangement in Figure 5.9 has more images on one side of the folio since the parchment on that side has a more complex shape.

Figure 5.9: A visualisation of the camera positions used to image a single folio side.

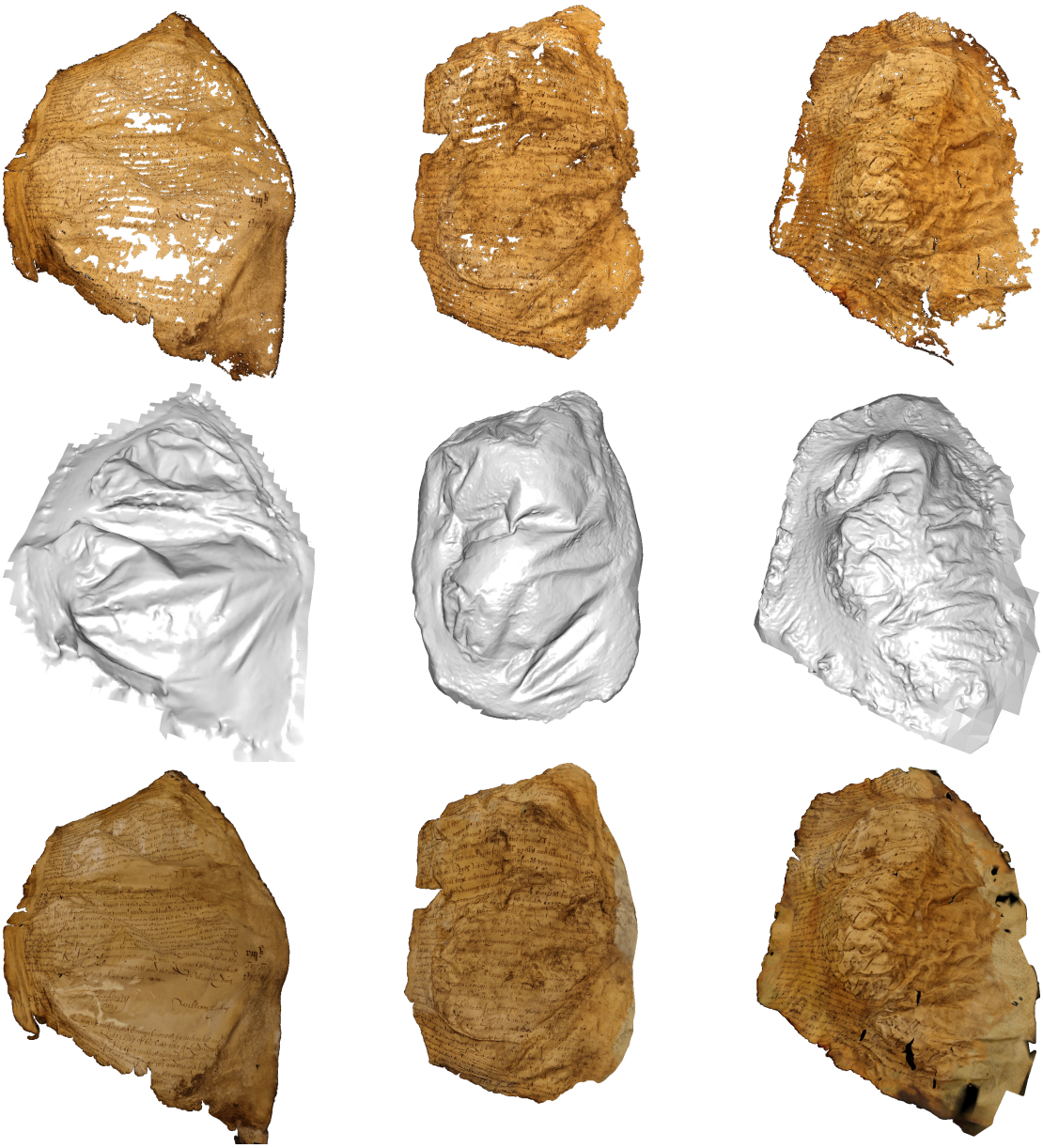

Figure 5.10: *Top*: Dense point clouds generated using VisualSFM and PMVS. *Middle*: Triangle meshes generated by applying Poisson Surface Reconstruction to the point clouds. *Bottom*: Textured meshes generated by back-projecting and blending the original images.

Figure 5.10 shows various textured reconstructions of parchment folios generated by our pipeline.

Figure 5.12 shows the histograms and heatmaps generated for eight different reconstructions of folios of the Great Parchment Book. From these we can see that the almost all of the parchments' surfaces are imaged at greater than 600 DPI, all with low sampling anisotropy. Typically, the areas of lower DPI lie on the borders of the parchment where there is no text and they were,

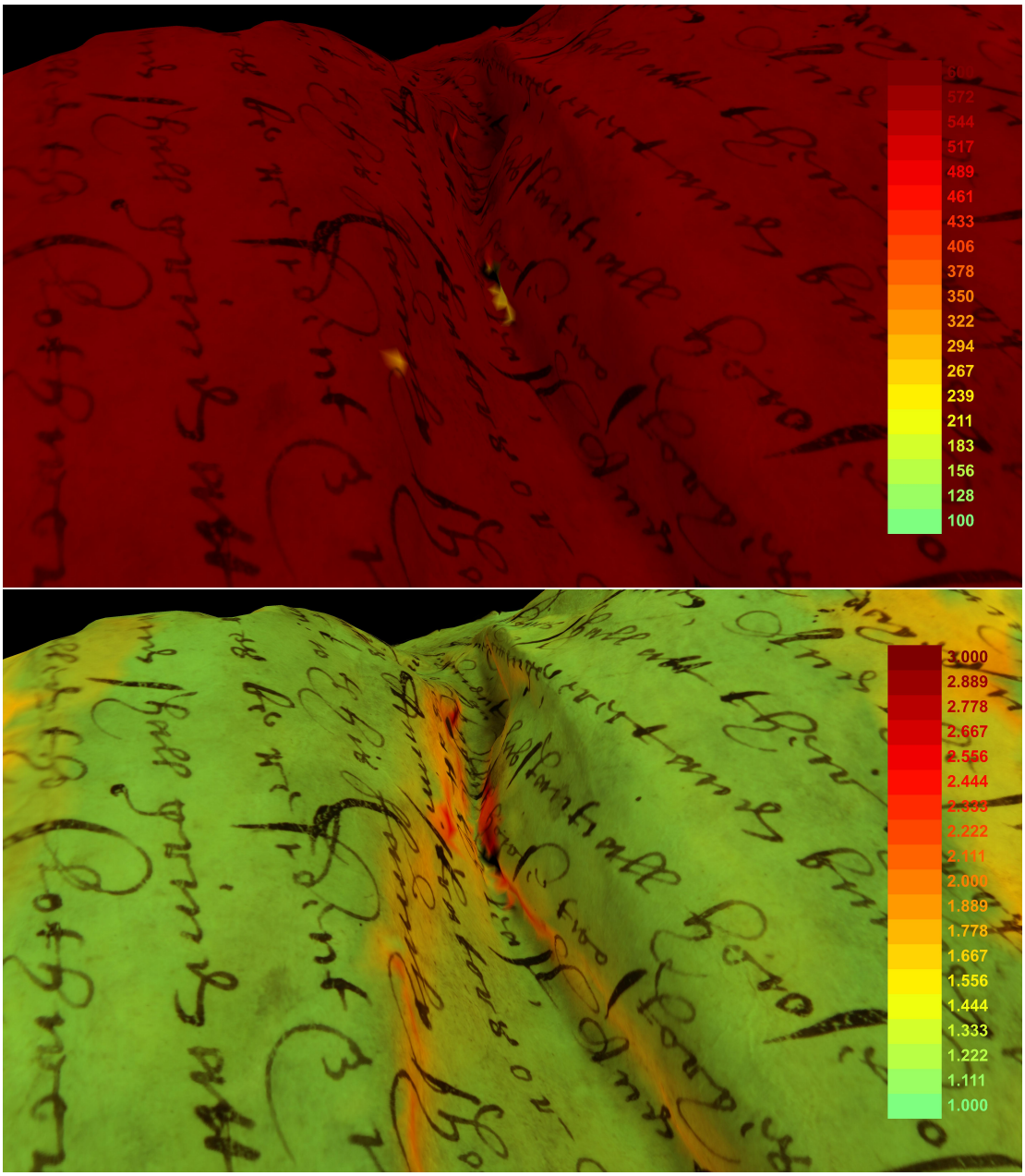

Figure 5.11: *Top*: Heat-map of sampling rate, *Bottom*: and sampling anisotropy inside a deep crease of a parchment.

therefore, imaged with a lower density. Area's of high stretch typically lie on heavily slanted areas and inside creases where a fronto-planar image would have been hard to obtain. Such a crease is shown in Figure 5.11. Despite being inside a deep crease, we can see that this area of the folio was still imaged with a high sampling density.

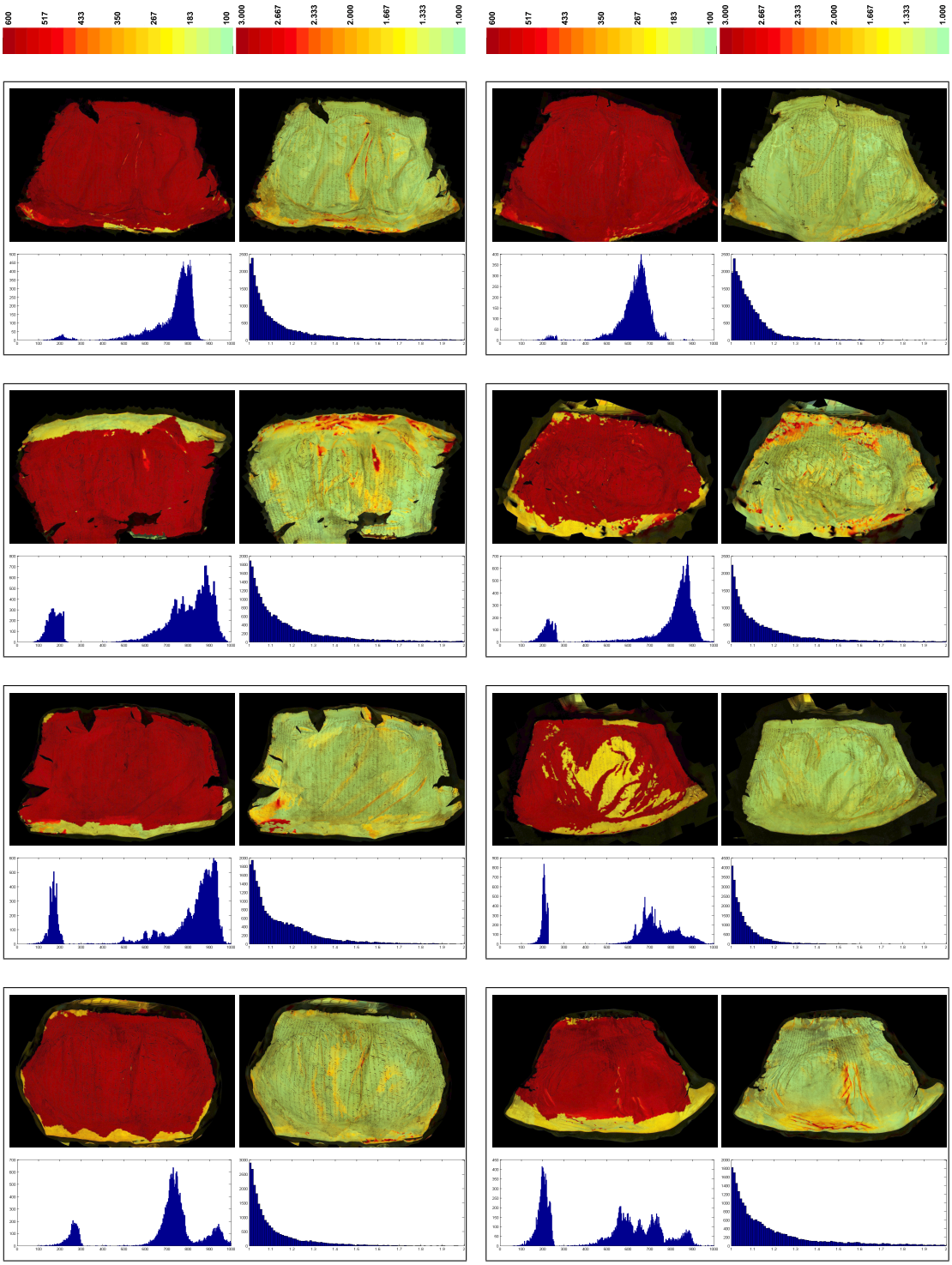

Figure 5.12: Within each sub-figure: *Top-Left*: DPI heatmap, *Top-Right*: Anisotropy heatmap, *Bottom-Left*: DPI histogram, *Bottom-Right*: Anisotropy histogram.

5.6 Discussion

We have proposed a pipeline for imaging and reconstruction of historical documents, particularly those with complex geometric distortions which would be difficult to capture using standard document scanning methods.

Our imaging approach uses a single hand-held camera to capture a set of overlapping images which fully captures the document surface. This approach gives the flexibility to adapt the imaging procedure to a wide variety of document surface shapes, in a way that a fixed camera arrangement cannot. Since the process does not require specialist equipment, such as structured-light or laser scanners, or large imaging rigs containing many cameras, it could be easily adopted by other archives or museums who are likely to have access to a high-resolution SLR camera.

While this imaging approach is simple, flexible, and easy to adopt, it does present a number difficulties. The greatest problem is that it is possible to capture a low-quality image set, with inadequate coverage or containing blurry images, and not know until the 3D reconstruction has been generated. The responsibility for image-set quality falls on the person taking the images. In Chapter 8 we discuss potential extensions to our imaging process which could at least partially automate this quality assurance.

The reconstruction method generates a high resolution textured 3D model of the document, meeting quality requirements of professional conservation and archival applications. These 3D models can be used purely for archival purposes or can serve as input for the virtual restoration methods which will be discussed in Chapters 6 and 7 It is important to note, however, that the virtual restoration approaches are not dependent on this specific acquisition process, which can be adapted to reflect the requirements of specific projects, different time and monetary budgets, and the development of new technologies.

We also proposed an intuitive visualisation of the spatially-varying sampling density, providing a means to gauge data quality with respect to archival standards. Since a single DPI value is insufficient for a 3D reconstruction, we propose that plotting histograms of the sampling density distribution allows us to see how much of a given folio has been imaged at a given effective DPI. The

histogram provides a way to judge, at a glance, the quality of the image set.

Chapter 6

Interactive Document Exploration

In Chapter 4 we discussed how standard metric-preserving surface parameterization algorithms are often not suitable for flattening parchment, due to the nature of parchment as a material and how it deforms when exposed to heat and/or moisture.

In this chapter we present an interactive system which circumvents these difficulties by allowing a user to navigating the surface of a 3D reconstruction of a document. At all times the system presents a locally flattened view of the region of text that the user is currently focussed on. Our system aims to improve the accessibility and legibility of text in highly distorted documents, in a manner which does not require a global parameterization. This approach is inspired by the observation that, when transcribing a text, a palaeographer will only ever inspect a small section of a folio at any given time [Youtie, 1963]. It is, therefore, unnecessary to un-distort the entire folio at once if the primary goal is to simply expose the content in a form that can be read.

Our system also addresses the issue of *provenance*. For historians studying the text through a digital representation, it is important to be able to judge whether a feature present in the digital representation was also present in the original text or whether is it an artefact of the reconstruction pipeline. Terras [Terras, 2011] discusses this issue at length, focussing mainly on imaging artefactsn and the London Charter for the Computer-based Visualisation of Cultural Heritage[Denard, 2012] stressed the importance of storing paradata which documents the process of generating the visualization. In our case the most likely source of error is the 3D reconstruction process. We therefore document the

provenance of the reconstruction by providing the user with smart access to the original image collection. By comparing the 3D reconstruction with the original images the user can better assess the content of the text in areas of the 3D reconstruction which seem to contain errors.

The viewer can also explore arbitrary 3D models, to inspect interesting surface details on objects for the purpose of, for example, archaeological or forensic examination.

6.1 Approach

We adopt the approach of undistorting local subsets of the mesh. Local regions are typically more approximately planar and have a lower surface-area to perimeter ratio and will thus undergo very little stretch distortion, and since they are flattened independently they will not be affected by reconstruction artefacts elsewhere in the mesh. Lastly, we can more easily rectify these local patches so that the text always runs from left to right without having to compute a consistent global distortion field.

We can make an analogy to the problem of map making. The global-flattening approach is equivalent to unwrapping the entire globe onto a single map of the earth which will contain the distortions typical of large map projections [Snyder, 1993]. Our approach is equivalent to navigating over the 3D globe while creating numerous local maps as we go, each of which contains low levels of distortion.

The goal of our viewer is that, as the user looks at a particular region (the target region) of the mesh, it should be displayed in a way that is optimal in terms of its readability. This optimality can be defined in terms of three sub-goals:

1. **Visibility**: The text in the region should all be visible so that the reader can properly take account of the context.

2. **Clarity**: The text should not appear distorted.

3. **Screen-Alignment**: Lines of text should be rectified with the screen so that it runs horizontally from left to right, and navigation over the parchment

should follow the lines.

We propose two modes of undistortion to solve goals (1) and (2):

1. **Local-affine**: the mesh is rendered in 3D, transformed so that the target region is oriented to face the camera.

2. **Local-flattening**: the target region is flattened into 2D (independently of the rest of the mesh).

In order to obtain an estimate of the physical warp due to damage, we employ a pre-processing step in which we compute a vector field V defined on the mesh surface which flows parallel to the lines of text. We use V to solve goal (3). We can rectify the text since V encodes the local orientation of the text at each point, and can navigate along lines of text by moving through the flow lines of V. Moving horizontally on screen corresponds to moving parallel to V, and moving vertically on screen to moving perpendicular to V.

To allow interactive speeds, we use the local-affine mode as the user pans over the mesh (essentially a preview of the local-flattening), and the local-flattening mode when the user stops panning. This interaction sequence is illustrated in Figure 6.1.

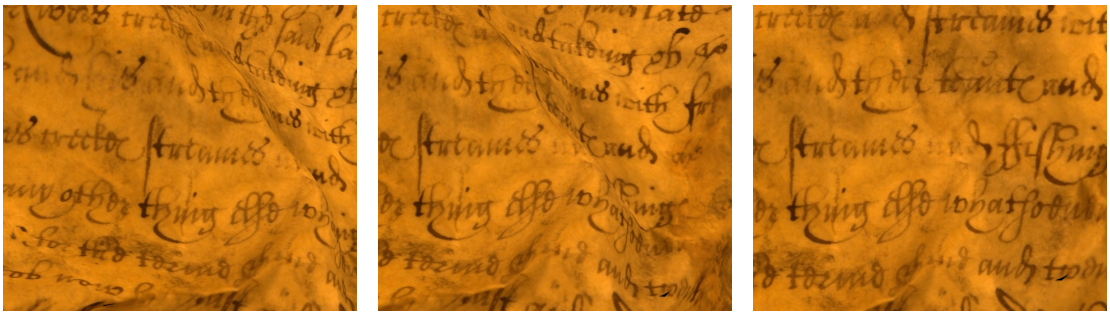

Figure 6.1: The *local-affine view* and *local-flattening* modes. Between *Left* and *Middle* the user pans slightly to the right. In *Right* the user stops panning and local-flattening is performed, removing perspective distortions and revealing otherwise hidden text.

6.1.1 Interactive Viewer

Our viewer application is somewhat inspired by interactive map viewers such as Google Earth, in that it allows a user to explore a three-dimensional surface in a way that makes the surface appear flat.

In the case of a map-viewer, the earth is spherical and navigation around the earth corresponds to moving along lines of latitude and longitude. However when sufficiently zoomed in, the map appears flat, without any of the distortions commonly associated with full map projections, and as the user pans over the map they appear to be panning over a two-dimensional surface.

In the case of our parchment-viewer, the lines of "latitude" and "longitude" are defined by the lines of text, with "latitude" equivalent to lines of text, with the lines of latitude flowing parallel to the text and the lines of longitude perpendicular to it.

6.1.2 Pre-Processing

We first pre-compute these lines of latitude and longitude. To do this, we select k sample points $\mathcal{S} = \mathbf{s_1}...\mathbf{s_k}$, randomly distributed over the surface of the mesh $\mathcal{M}$, by selecting k random vertices on the mesh. The value of k can be set to be equal to the total number of vertices; however, we find that setting k equal to one tenth of the number of vertices works sufficiently well in most cases and significantly reduces the amount of processing required. For each sample point $\mathbf{s}_i$, which has a surface normal $\mathbf{n}_i$, we first define the zero-rotation viewing matrix M_i^0 as follows.

We pick a dummy right-vector $\mathbf{v}_{\text{temp}}$ equal to $(1,0,0)^T$ (unless $\mathbf{n}_i$ equal to $(1,0,0)^T$, in which case we set $\mathbf{v}_{\text{temp}}$ equal to $(0,0,1)^T$). Then, the zero-rotation up and right vectors $\mathbf{v}_{i,\text{up}}^0$ and zero-rotation up-vector $\mathbf{v}_{i,\text{right}}^0$ at $\mathbf{s}_i$ are given by:

$$\mathbf{v}_{i,\text{up}}^0 = \mathbf{n}_i \times \mathbf{v}_{\text{temp}}, \tag{6.1}$$

$$\text{and} \quad \mathbf{v}_{i,\text{right}}^0 = \mathbf{v}_{i,up} \times \mathbf{n}_i, \tag{6.2}$$

Now $\mathbf{n}_i$, $\mathbf{v}_{i,right}^0$, and $\mathbf{v}_{i,up}^0$ form an orthonormal coordinate frame with its origin at $\mathbf{s}_i$, with one axis normal to the surface and the other two axes in the tangent plane of the surface. Next, we need to compute the angle, θ_i, such that rotating this coordinate frame about $\mathbf{n}_i$ will align $\mathbf{v}_{i,right}^0$ with the direction of the text. To do this we then extract a subset, $\mathcal{M}_i$, of $\mathcal{M}$ consisting of all the triangles which lie completely within a radius r of $\mathbf{s}_i$, and then apply Least-Squares-

Conformal-Mapping (LSCM) [Lévy et al., 2002] to $\mathcal{M}_i$ to obtain a flattened sub-mesh $\mathcal{M}_i^F$. We then render $\mathcal{M}_i^F$ into an image and then find the rotation which properly aligns the text inside this patch.

We use the insight that, for a given patch of the document image which has been approximately segmented into text and background regions, the horizontal projection of the patch will exhibit a characteristic pattern when the patch is rotated such that the lines of text are horizontal. An example of this is shown in Figure 6.2. We can see that when the lines of text are misaligned, the distribution of the horizontal projection is fairly uniform, but as the lines are rotated to be well-aligned, the horizontal projection contains sharp peaks coinciding with the vertical positions of the lines and troughs coinciding with the inter-line whitespace. This characteristic shape can be measured by considering the variance of the distribution. Specifically, the distribution should exhibit a maximum variance when the patch is rotated to make the lines of text horizontal.

Given a flattened patch, $\mathcal{M}_i^F$, we approximately segment text from background using Otsu's method [Otsu, 1975] to select an intensity threshold, giving us a segmented image patch P_{seg}. This segmentation need only be approximate, since it only needs to capture the general location of a text line. We then search for the optimal rotation angle in increments of one degree between maximum and minimum possible rotations $\theta_{\min}$ and $\theta_{\max}$ (which are pre-selected by a user as sensible bounds based on an inspection of the parchment). For each possible angle $\theta \in [\theta_{\min}, \theta_{\max}]$ we rotate the P about its center by θ to generate the rotated patch P_θ, and then compute the horizontal projection vector $\vec{h}_\theta$ by summing along each row of P_θ. We then compute the variance var_θ of $\vec{h}_\theta$ and if it is the highest so far, we store θ and var_θ.

Letting R_i be the 3D rotation matrix which performs a rotation about $\mathbf{n}_i$ by θ_i, we then construct the final text-aligned coordinate frame at $\mathbf{s}_i$ with axes $\mathbf{n}_i$, $\mathbf{v}_{i,right}$ (which runs parallel to the text), and $\mathbf{v}_{i,up}$ (which runs perpendicular to

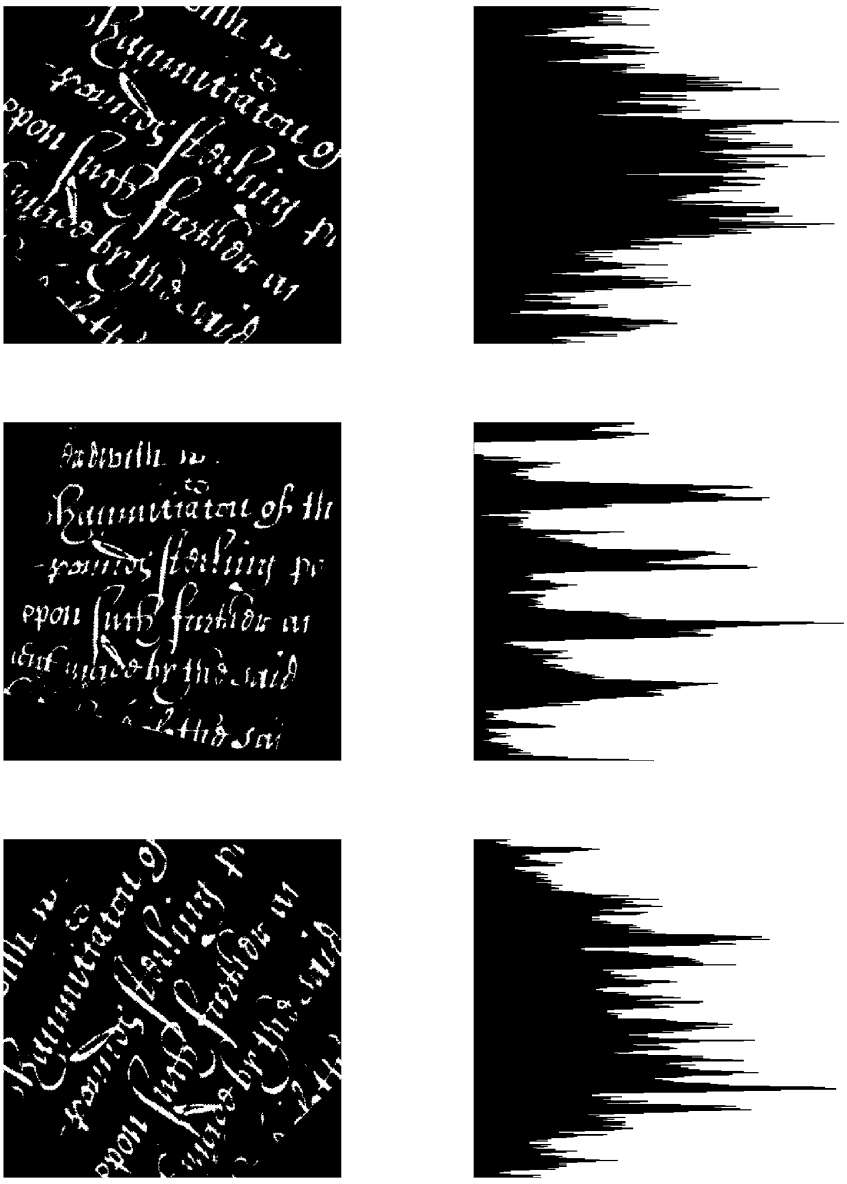

Figure 6.2: *Left*: A patch of text at various rotations. *Right*: The horizontal projection for each patch. The y-axis of these plots corresponds to the pixel row of the associated patch, and the x-axis to the number of text pixels in that row. Notice that, as the distribution of the horizontal projection shows stronger peaks and higher variance, the text lines are more closely aligned with the horizontal axis.

the text), where:

$$\mathbf{v}_{i,right} = R_i \mathbf{v}^0_{i,right}, \tag{6.3}$$

$$\text{and } \mathbf{v}_{i,up} = R_i \mathbf{v}^0_{i,up}, \tag{6.4}$$

giving us the vector-field $V = \mathbf{v}_{i,right}$ which is defined on the surface of the mesh and flows parallel to the lines of text. We then smooth the V by applying a Gaussian kernel over the mesh, and interpolate the values of V within triangles using barycentric coordinates.

6.1.3 Surface Exploration

We can now V to allow the user to navigate the mesh surface in accordance with the lines of text. At any time τ during the execution of the viewer, we keep a view-target point $\mathbf{p}^\tau$, which is a point on the surface of the mesh $\mathcal{M}$ on which the camera will be centred. Associated with $\mathbf{p}^\tau$ are the the surface normal $\mathbf{n}^\tau$ (as illustrated in Figure 6.3 (a)), a right-vector $\mathbf{v}^\tau_{\text{right}}$ in V, and up-vector $\mathbf{v}^\tau_{\text{up}}$ perpendicular to $\mathbf{v}^\tau_{\text{right}}$ and also in the tangent-plane of the surface at $\mathbf{p}^\tau$.

The user is able to click and drag to move $\mathbf{p}^\tau$ over the mesh surface. If in a given time-step the user drags the mouse by some step $\mathbf{d} = (d_x, d_y)^T$ measured in pixels in screen-space, we compute the update-step vectors:

$$\mathbf{ffi}_{\text{right}} = d_x \mathbf{v}_{\text{right}}, \tag{6.5}$$

$$\text{and} \quad \mathbf{ffi}_{\text{up}} = d_y \mathbf{v}_{\text{up}}, \tag{6.6}$$

and then compute an intermediate view-target, illustrated in Figure 6.3 (b), as:

$$\mathbf{q}^\tau = \mathbf{p}^\tau + \mathbf{ffi}^\tau,$$

$$\text{where} \quad \mathbf{ffi}^\tau = \mathbf{ffi}_{\text{right}} + \mathbf{ffi}_{\text{up}}, \tag{6.7}$$

Then to ensure that the new view-target is on the mesh surface, we construct a ray $R(t)$:

$$R(t) = \mathbf{q}^\tau + t\mathbf{n}^\tau, \tag{6.8}$$

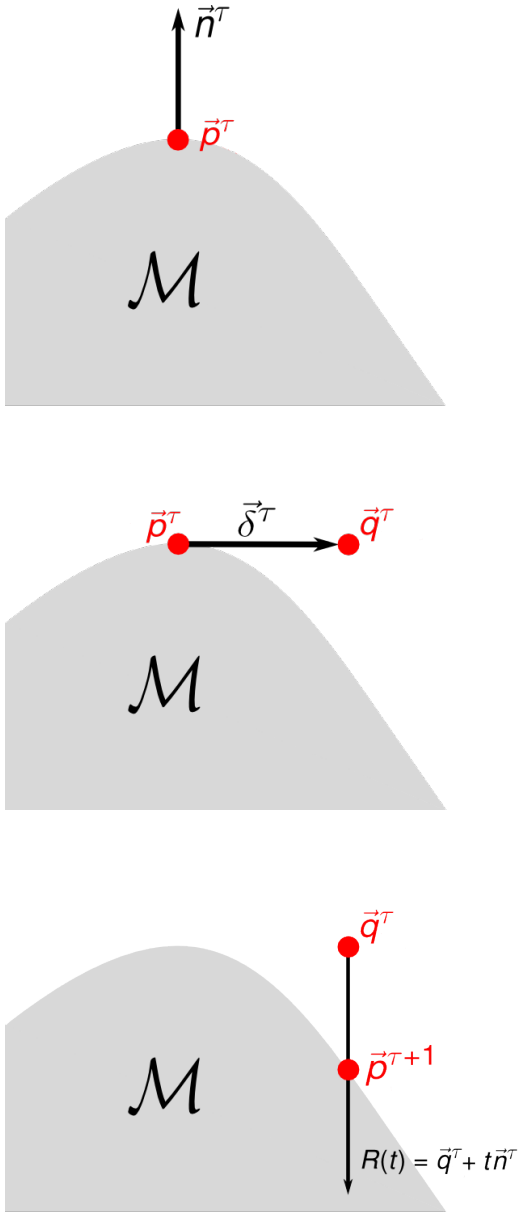

Figure 6.3: The view-target update steps

which intersects the mesh surface at some value t_S of the distance parameter t. Now we compute the actual updated view-target, illustrated in Figure 6.3 (c), as:

$$\mathbf{p}^{\tau+1} = \mathbf{q}^\tau + t_S \mathbf{n}^\tau, \tag{6.9}$$

For any given view-target $\mathbf{p}$, on the surface of the mesh, with right-vector $\mathbf{v}_{\text{right}}$ and up-vector $\mathbf{v}_{\text{up}}$, we compute the appropriate OpenGL model-view matrix (the viewing matrix) such that the mesh will be rendered with the lines of text around $\mathbf{p}$ flowing horizontally on the screen.

We construct a local coordinate frame on the surface of the mesh with origin $\mathbf{p}$ and $\mathbf{v}_{\text{right}}$ and $\mathbf{v}_{\text{up}}$ as axes. We also define a canonical coordinate frame as having origin $\mathbf{q} = (0,0,0)^T$, and right and up axes as $(c_{\text{zoom}},0,0)^T$ and $(0,c_{\text{zoom}},0)^T$ respectively. The value c_{zoom} represents the current zoom level. If it is large, then the canonical frame will occupy a larger area, and the un-warped text will appear larger. These two coordinate frames are illustrated in Figure 6.4. We then compute the 3D affine transformation matrix A which maps the surface coordinate frame onto our canonical coordinate frame as follows.

First, let:

$$\begin{aligned} \mathbf{p}_{\text{right}} &= \mathbf{p} + \mathbf{v}_{\text{right}}, & \mathbf{q}_{\text{right}} &= \mathbf{q} + (c_{\text{zoom}},0,0)^T, \\ \mathbf{p}_{\text{up}} &= \mathbf{p} + \mathbf{v}_{\text{up}}, & \mathbf{q}_{\text{up}} &= \mathbf{q} + (0,c_{\text{zoom}},0)^T, \\ \mathbf{p}_{\text{cross}} &= \mathbf{v}_{\text{right}} \times \mathbf{v}_{\text{up}}, & \mathbf{q}_{\text{cross}} &= (c_{\text{zoom}},0,0)^T \times (0,c_{\text{zoom}},0)^T, \end{aligned} \tag{6.10}$$

Then, construct 4×4 matrices T_1 and T_2 as:

$$T_1 = \begin{bmatrix} \mathbf{p} & \mathbf{p}_{\text{right}} & \mathbf{p}_{\text{up}} & \mathbf{p}_{\text{cross}} \\ 1 & 1 & 1 & 1 \end{bmatrix}, \tag{6.11}$$

$$T_2 = \begin{bmatrix} \mathbf{q} & \mathbf{q}_{\text{right}} & \mathbf{q}_{\text{up}} & \mathbf{q}_{\text{cross}} \\ 1 & 1 & 1 & 1 \end{bmatrix}, \tag{6.12}$$

Since A is the transformation which maps the columns of T_1 onto the columns of T_2, we have

$$\begin{aligned} AT_1 &= T_2, \\ A &= T_2 T_1^{-1}, \end{aligned} \tag{6.13}$$

Setting A as the OpenGL model-view matrix will cause the mesh to be rendered with the lines of text in the vicinity of $\mathbf{p}$ aligned with the canonical coordinate frame, that is flowing horizontally along the screen. Currently, we always construct the surface coordinate frame to be orthonormal, i.e. $\mathbf{v}_{\text{right}}$ and $\mathbf{v}_{\text{up}}$ are both unit-length and orthogonal to one-another. However, if in the future we adapt our text-flow detection procedure to detect not only the orientation of

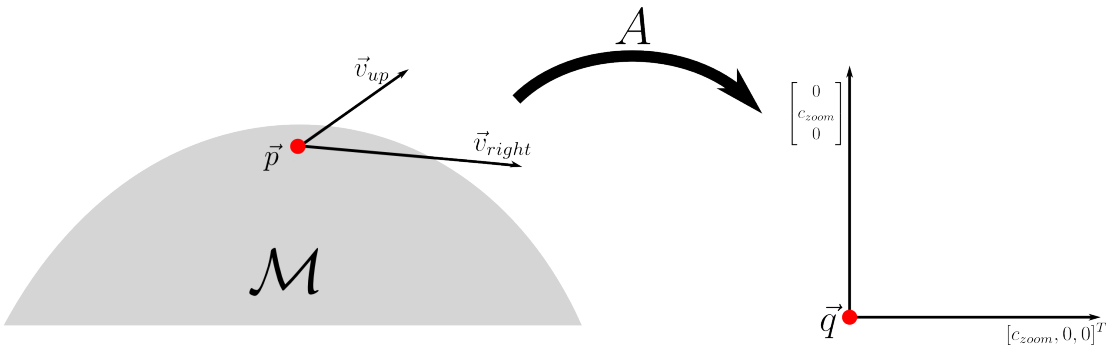

Figure 6.4: Left: The local coordinate frame on the surface of the mesh, describing the local orientation of the text. Right: The canonical coordinate frame. The mapping between these two coordinate frames is the affine transform A.

the text but also the its scale and any shearing distortions, these could also be encoded in the surface coordinate frame by using non-unit-length (to encode scale) and non-orthogonal (to encode shear) axes. Since we model the local warp from surface frame into canonical frame as a general affine transformation, such extensions would be handled automatically.

6.1.4 Local Conformal Flattening

As the user pans around the parchment model, the viewing matrix is constantly updated as above so that panning left and right moves the camera along lines of text and panning up and down move the camera perpendicular to lines of text. However our application performs a second kind of un-warping when the user stops panning and settles on a target point **p**. We extract a subset of the mesh consisting of all the triangles which lie completely within a radius (defined by the current field of view) of **p**, and then flatten this sub-mesh using Least-Squares-Conformal-Mapping (LSCM) [Lévy et al., 2002], described in full in Section 2.5.

In our implementation we use OpenNL in conjunction with SuperLU [Demmel et al., 1999] for numerically solving linear systems.

The result of this LSCM un-warping is a flattened mesh, $\mathcal{M}^F$, with some arbitrary rotation. In order to maintain a sense of continuity between the un-flattened and flattened views of the mesh, we perform a simple registration step to align the flattened mesh with the un-flattened mesh from which it was generated.

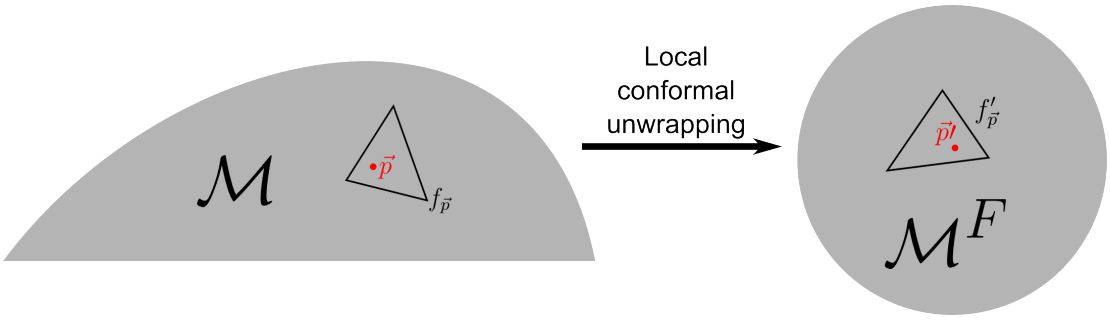

Figure 6.5: A local patch of the mesh $\mathcal{M}$ around the viewing target $\mathbf{p}$ (left) is unwrapped into a flattened mesh $\mathcal{M}^F$ (right) centred on the corresponding point $\mathbf{p}'$

We define $f_{\mathbf{p}}$ to be the triangle in the original mesh containing the viewing target $\mathbf{p}$, $f'_{\mathbf{p}}$ to be the corresponding triangle in the flattened mesh, and $\mathbf{p}'$ to be the point in $f'_{\mathbf{p}}$ having the same barycentric coordinates as $\mathbf{p}$ in $f_{\mathbf{p}}$. This correspondence is illustrated in Figure 6.5.

We then register the unfolded patch with the original mesh by aligning $f'_{\mathbf{p}}$ with $f_{\mathbf{p}}$ using an affine transformation.

6.1.5 Data Provenance

In addition to this model-explorer aspect of our application, we also took this opportunity to address another issue important in all digital cultural heritage applications, that being the issue of provenance. The provenance of an object refers to its entire history, including the groups or organisations which originally created it, its method of creation, its subsequent chain of custody and any modifications made to it. Since we are dealing with digital representations, we are also interested in the provenance of the digital data, that being the original image set and the subsequent processing steps which are applied to it. For historians studying the text (or any object for that matter) through a digital representation, it is important to be able to judge whether a feature present in the digital representation was also present in the original text, or whether is it an artefact of the digitization and processing pipeline. This issue is discussed at greater length by Terras [Terras, 2011], who discusses how "*if we cannot trust our means of reproduction of images of texts, can we trust the readings from them?*" and "*how do scholars acknowledge the quality of digitised images of texts?*" Arnold [Arnold, 2008] also discusses how "*a digital representation of an artefact is*

a representation of certain relevant characteristics of the artefact. It is not the original and complete artefact, nor even a metonymy or simulacrum of the complete artefact. It is only a representation of some 'relevant characteristics'."

Both of these authors are discussing errors introduced when a document is imaged, how the resulting image must, therefore, not be taken as a perfectly truthful representation of the document, and how a historian would ideally have some way of assessing the truthfulness of a digital representation. This issue is even more pertinent when the digital representation is a 3D reconstruction which is the result of further processing of the images. The only way to know for certain which features were or weren't present in the original text is to have the original text present, since artefacts can be introduced as soon as the digitization process begins. However, one of the primary reasons for having digital representations is that they can be accessed remotely without having the original text present, and so the next best option is to provide the user access to the original data used to construct the digital representation, effectively allowing the user to look back through the digital history of the text. In our case, these data are the images used to perform the 3D reconstruction.

Simply providing the user with the entire image set would be unsatisfactory. It would be too awkward and time-consuming for the user to have to manually search through a large unordered collection of images for a good view of a specific section of the text. Instead we provide a feature in our viewer application where the user is able to click on a point on the 3D model of the parchment, and our application will fetch an image from the original image set which sees that point well, rotate and scale it so that it is registered with current view of the 3D model, and display it to the user. This way, if the user is exploring a parchment folio and sees a unusual feature they can simply click on it to see whether that feature was present in the original image set or whether it is an erroneous artefact of the reconstruction process. An example of the usefulness of this feature is shown in Figure 6.9. Here we see that the writing circled in red looks slightly unusual, perhaps due to badly reconstructed geometry or being textured from a badly aligned camera. However, by inspecting that same section of the parchment as seen in one of the original images, we can see that

the text in question does in fact appear the same on the parchment, and is not the result of a bad reconstruction.

In our implementation of this provenance feature, we select which input camera best sees a point based on the same weighting scheme used for computing the camera blend fields for texture map generation, as described in Section 5.3.2. For a vertex v of the mesh, we pre-compute the camera $C(v)$ which sees it best, that is the camera which has the highest weight evaluated at v_i. To extend this to any point $\mathbf{p}$ on the mesh surface, we define the best camera $C(\mathbf{p})$ for $\mathbf{p}$ to be $C(v_p)$, where v_p is the nearest mesh vertex to $\mathbf{p}$. The visibility-erosion step in our weighting scheme ensures that the all the triangles incident on a vertex v are completely visible in $C(v)$, therefore a point $\mathbf{p}$ is guaranteed to be visible in $C(\mathbf{p})$.

Now when a user clicks on a point $\mathbf{p}^S$ in screen-space coordinates, we back-project this point onto the mesh using the current OpenGL camera matrices to obtain a 3D point $\mathbf{p}^M$ on the surface of the mesh. We then find the original camera $C(\mathbf{p}^M)$ in which $\mathbf{p}^M$ is best visible and register its image as best possible with the view of the 3D parchment model. Since the goal of this feature is to provide the user with access to the original data with minimal processing applied to it, we allow only rotation and scaling (i.e. a similarity transform) of the image to perform this registration.

Letting f_p^M denote the triangle in the mesh which contains $\mathbf{p}^M$, we project f_p^M into the OpenGL viewing camera to obtain the corresponding triangle f_p^S in screen-space coordinates, and into $C(\mathbf{p}^M)$ to obtain the triangle f_p^C in image pixel coordinates. Now, if we define our original-image viewing-window to have width and height of w_{view} and h_{view} respectively, we construct a third target triangle f_p^T as:

$$f_p^T = f_p^S - \mathbf{p}^S + \begin{bmatrix} 0.5w_{view} \\ 0.5h_{view} \end{bmatrix}, \tag{6.14}$$

The target triangle f_p^T is simply a translation of f_p^S to align $\mathbf{p}^S$ with the center of the original-image viewing-window. We then use Procrustes analysis to find the similarity transform $A_{T\to C}$ which best maps f_p^T onto f_p^C. We denote by $\mathbf{u}_0$,

$\mathbf{u}_1$, and $\mathbf{u}_2$ the vertices of f_p^C, and by $\mathbf{u}'_0$, $\mathbf{u}'_1$, and $\mathbf{u}'_2$ the vertices of f_p^T. We also denote the individual coordinates of $\mathbf{u}_i$ as x_i and y_i, and of $\mathbf{u}'_i$ as x'_i and y'_i. Then, we define the values:

$$\begin{aligned} S_x &= \sum_{i=0}^{2} x_i, & S_y &= \sum_{i=0}^{2} y_i, \\ S_{x'} &= \sum_{i=0}^{2} x'_i, & S_{y'} &= \sum_{i=0}^{2} y'_i, \\ S_{xx} &= \sum_{i=0}^{2} x_i^2, & S_{yy} &= \sum_{i=0}^{2} y_i^2, \\ S_{xx'} &= \sum_{i=0}^{2} x_i x'_i, & S_{yy'} &= \sum_{i=0}^{2} y_i y'_i, \\ S_{xy'} &= \sum_{i=0}^{2} x_i y'_i, & S_{x'y} &= \sum_{i=0}^{2} x'_i y_i, \end{aligned} \tag{6.15}$$

The transformation $A_{T\to C}$ has the form:

$$A_{T\to C}(\mathbf{x}) = \begin{bmatrix} a & -b \\ b & a \end{bmatrix} \mathbf{x} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}, \tag{6.16}$$

We compute the parameters, (a, b, t_x, t_y), of the transformation by solving the the linear system:

$$\begin{bmatrix} S_x & -S_y & n & 0 \\ S_y & S_x & 0 & n \\ S_{xx}+S_{yy} & 0 & S_x & S_y \\ 0 & S_{xx}+S_{yy} & -S_y & S_x \end{bmatrix} \begin{bmatrix} a \\ b \\ t_x \\ t_y \end{bmatrix} = \begin{bmatrix} S_{x'} \\ S_{y'} \\ S_{xx'}+S_{yy'} \\ S_{xy'}-S_{x'y} \end{bmatrix}, \tag{6.17}$$

Now, $A_{T\to C}$ can be used to extract the section of the original-image and align it correctly with the rendering of the mesh. The relationship between $\mathbf{p}^S$, the mesh $\mathcal{M}$, the OpenGL viewing camera, the original-camera $C(\mathbf{p}^M)$, and the three triangles f_p^S, f_p^C, and f_p^T is illustrated in Figure 6.6.

The original images in our parchment datasets are each 22 MegaPixels in size. The exact image size may vary from dataset to dataset but we expect the images to always be of a very high resolution in order to generate accurate,

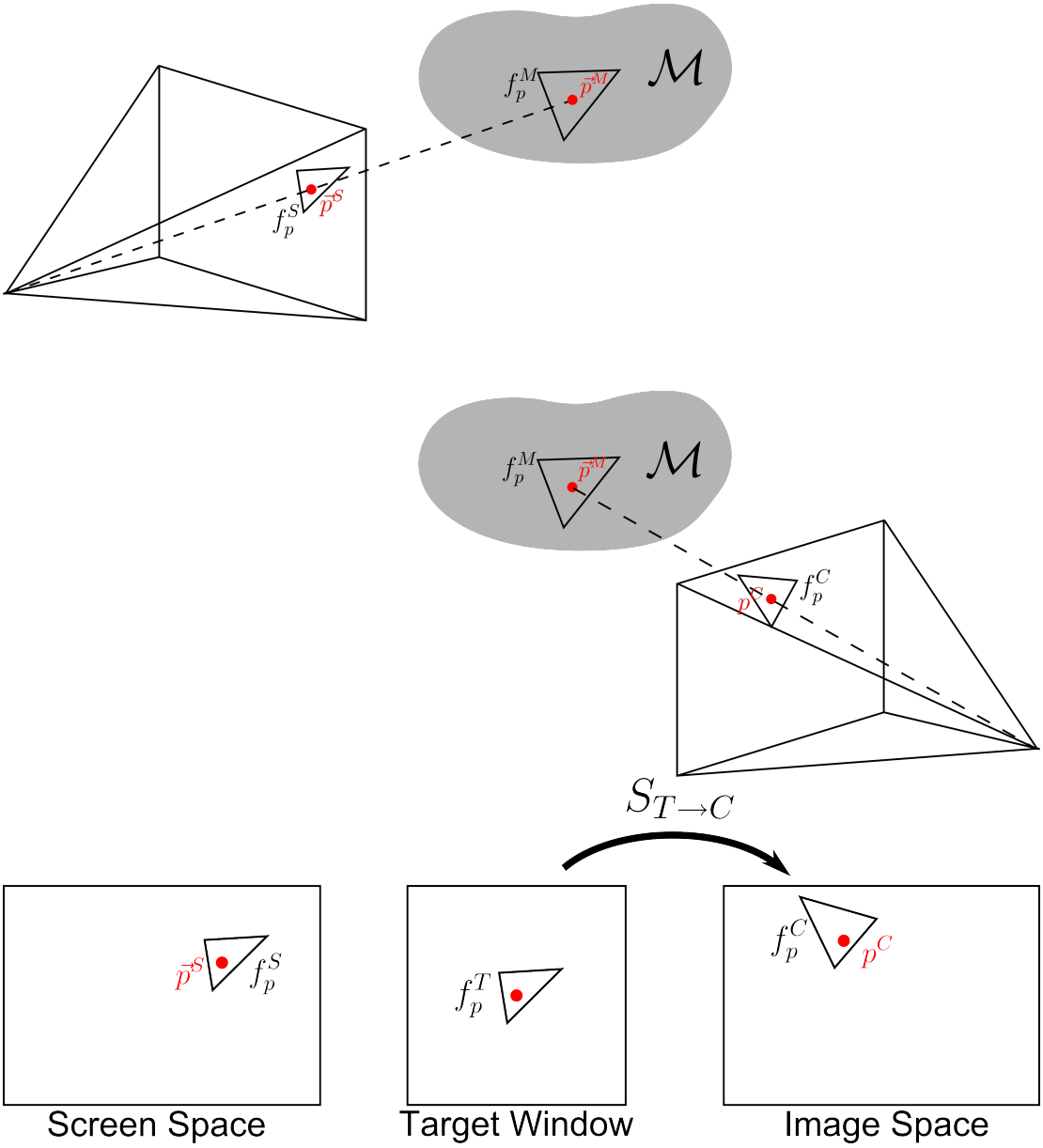

Figure 6.6: *Top*: The user clicks on a point $\mathbf{p}^S$ in screen-space, which is back-projected onto point $\mathbf{p}^M$ on the mesh. f_p^M is the triangle in $\mathcal{M}$ which contains $\mathbf{p}^M$ and f_p^S is the projection of this triangle back into the current OpenGL camera. *Middle*: $\mathbf{p}^C$ and f_p^C are the projections of $\mathbf{p}^M$ and f_p^M in the camera which sees $\mathbf{p}^M$ best. *Bottom*: f_p^T is the result of translating f_p^S such that $\mathbf{p}^S$ is centred in the target window. $S_{T\to C}$ is the similarity transform which best aligns f_p^T with f_p^C.

high resolution meshes. Loading an entire such image from disk each time the user uses the provenance feature is too slow to be acceptable for an interactive application. We therefore use a tiling engine to speed up this process.

For each image in the dataset, we break the image into a number of non-overlapping $\beta \times \beta$ tiles which we save to disk. Since we only ever need to access a small subset of an image at a time, instead of loading the entire image at once, we load only the tiles which intersect that section of the image which we are interested in.

6.2 Results

We show visualisation results using our application on reconstructions from two folios of the Great Parchment Book. The reconstructions consist of approximately 50,000 triangles, and the textures are in the order of 100 megapixels.

6.2.1 Local Flattening

Figures 6.7, and 6.8 show example results of our system in local-affine mode (left) and local-flattening mode (right). In the local-affine mode the text is correctly aligned in the viewing camera so that it runs from left to right, however distortions are still visible in very crumpled regions. In Figure 6.7e a large amount of text is completely obscured from view by a fold, in Figures 6.7a and 6.7c the text around the fold is not obscured but is distorted and foreshortened by perspective, and in Figures 6.8a and 6.8c, even though most of the text is visible, the distortions and occlusions caused by the sharp creases and fine wrinkles have a significant detrimental effect on legibility.

We can see how the local-flattening mode solves these problems. In Figure 6.7f the folio is unfolded and a large previously occluded chunk of text becomes visible, in Figures 6.7b and 6.7d, the foreshortening and perspective distortion is removed, and in Figures 6.8b and 6.8d the fine creases and wrinkles are opened out revealing some previously occluded text and significantly reducing the overall distortion.

6.2.2 Provenance

In Figure 6.9 , we can see two examples of the usefulness of the provenance

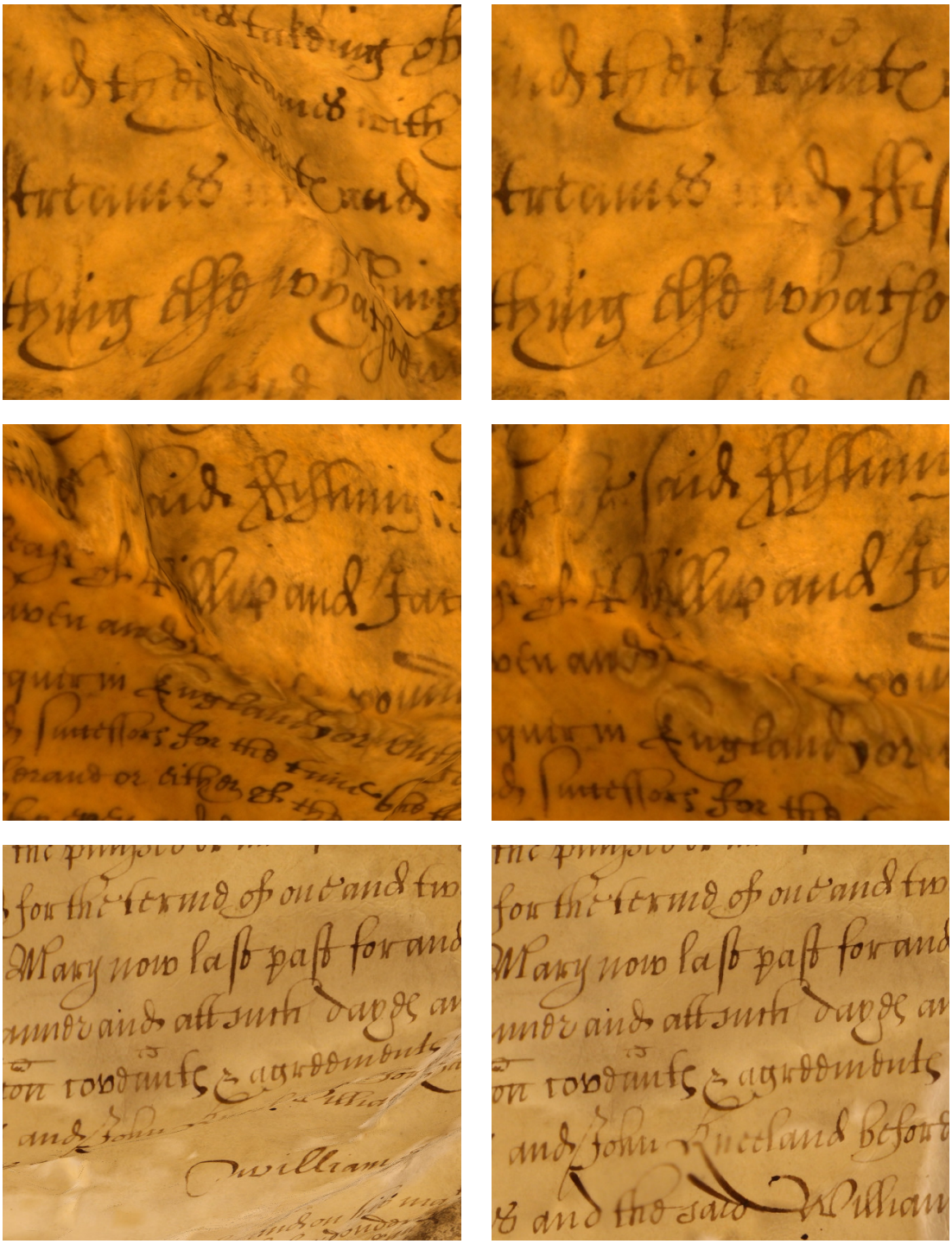

Figure 6.7: Sections of parchment folios rendered in *Left:* local-affine mode. *Right:* local-flattening mode.

feature. In Figure 6.9 (Top) we see a word which looks very unusual and could reasonably be thought to be an error introduced by the reconstruction process. By inspecting the same region of the folio as seen by one of the source images, we can see that the reconstruction is actually an accurate rendition of the text.

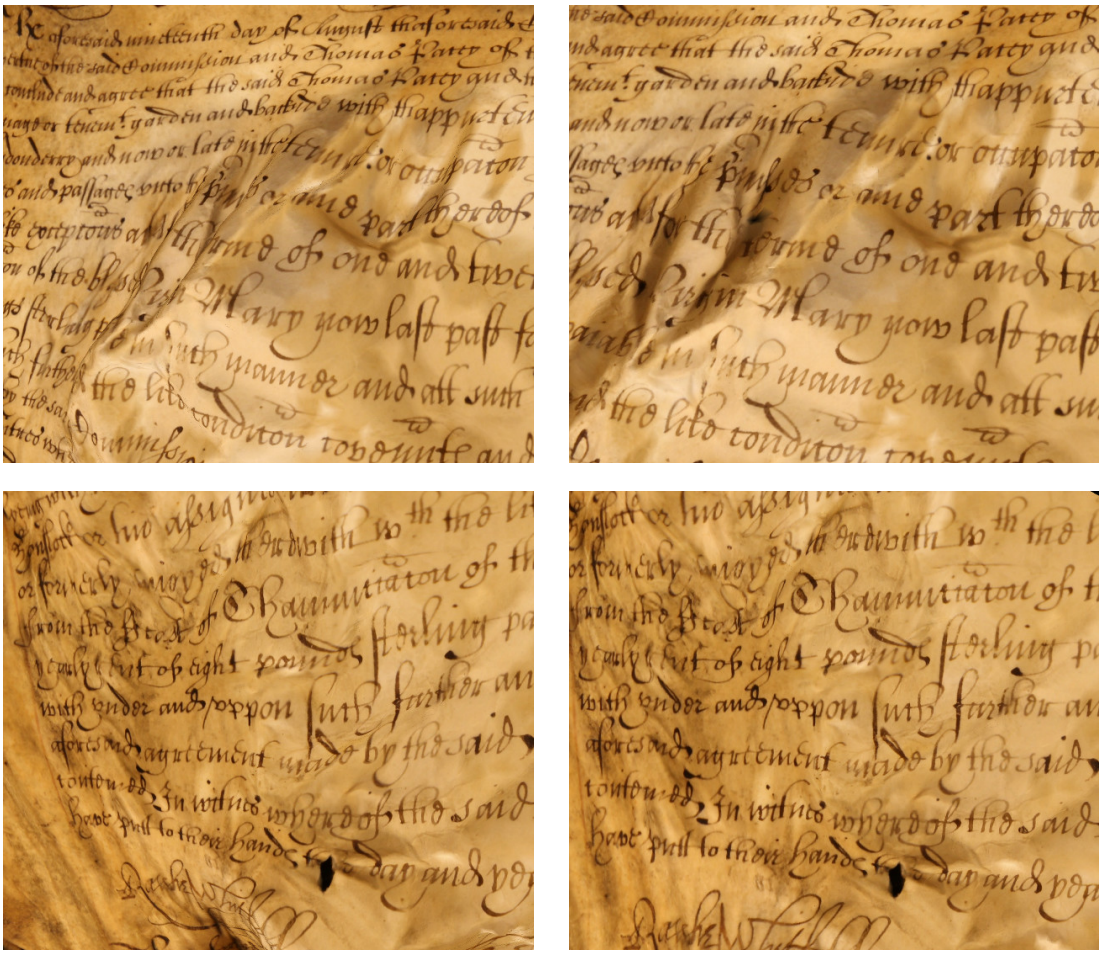

Figure 6.8: Sections of parchment folios rendered in *Left:* local-affine mode. *Right:* local-flattening mode.

In Figure 6.9 (Bottom) we see a word inside a crease which has become blurred during the reconstruction and texturing process to the point that it is illegible. By looking at the corresponding source image, the word can be seen more clearly and is legible.

6.2.3 User Evaluation

The key success metric for our system is the ease with which an expert palaeographer can read the text, specifically the confidence with which they can identify certain words and letters. This quality is difficult to measure, so we instead evaluated our system with three users with experience in transcribing medieval scripts. The users were asked to spend time transcribing texts presented in three modalities: a globally flattened static image, the viewer with only local-affine undistortion enabled, and the fully functional viewer. Static images were used

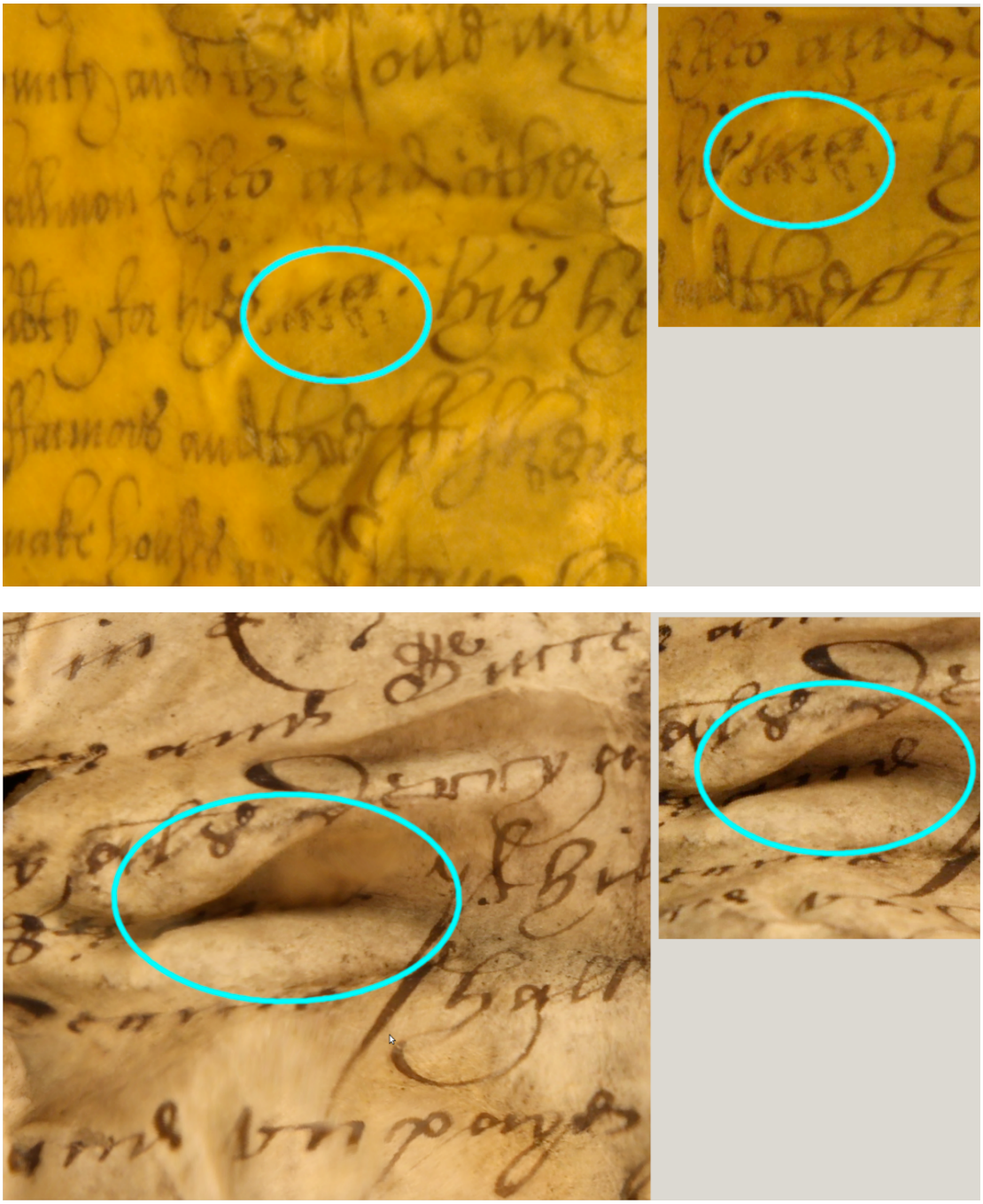

Figure 6.9: Examples of the provenance feature. The textured 3D mesh is displayed on the left, and the corresponding section of one of the original images on the right. *Top*: The markings in the view of the 3D mesh look unusual, as though they may be an erroneous artefact of the reconstruction process. However, inspection of the original image shows it to be an accurate representation of the parchment. *Bottom*: The text inside the crease has become blurred in the reconstruction process. However, inspection of the original image provides a clearer view of this text.

which did not contain large stretch distortions or distortions due to reconstruction artefacts. We then interviewed the participants to try to understand their experience using each modality.

We found that a static image is preferable in areas where the text was originally fairly undistorted since a global approach effectively flattens such areas, and the simple 2D pan-and-zoom style of interaction with images is simple and familiar. However in highly crumpled areas, the feedback indicated that text was harder to decipher in a static image, and being able to manipulate the folio in 3D was useful. One participant commented that they *"would not trust the [global] image on its own"* and that it was *"helpful to have the option of looking into creases"* in local-affine mode. One participant observed that the local-flattening mode allows for easier reading the majority of the time but that in certain crumpled areas, navigating back and forth over a crease allowed them to view a difficult word from a number of angles, mimicking the standard palaeographic method of physically manipulating the manuscript to observe that word from multiple viewpoints. They also pointed out that exploring a crumpled area in local-affine mode gave them *"a better sense of the level of damage"* which could then influence they way in which they transcribed that area in local-flattening mode.

Generally the participants indicated that having both modes available and being able to go back and forth between each was useful and allowed them to read the text with more confidence.

They also found the provenance feature to be useful for the reasons which we intended. The participants confirmed that it was useful for transcribing difficult areas where the reconstruction was unclear, and also because it allows the transcriber to "trust what they see" in the reconstruction.

6.3 Discussion

In this Chapter we have presented an interactive system for exploring highly distorted documents in which the user is always shown a locally flattened view of the region of the document they are inspecting. We have demonstrated that the current flattening methods can fail in cases of highly distorted documents

and proposed an alternative approach in which local regions of the text are undistorted dynamically. We have shown how this interactive, local flattening is preferable to static, global flattening for exploring documents with large distortions.

We proposed two modes of undistortion: a *local-affine* mode where the local region of text is aligned with the screen by an affine transformation, and a *local-flattening* mode which also flattens that local region by parameterising it in 2D. We have shown how the local-flattening mode is superior to the local-affine mode as it unfolds creases in the folio to expose text that is otherwise foreshortened or hidden, but also that both undistortion modes are useful in the transcription process when used in conjunction. The provenance feature was also show to be helpful for resolving ambiguities and giving the transcriber a greater sense of trust in the reconstruction.

Assuming a 3D reconstruction of sufficient quality is available, the interface (without flattening) provides equivalent access to the content of the text as direct access to document. With the flattening mode enabled the interface provides the capability of visualising the text in ways which are impossible with the physical document, and which in some cases improve the legibility of the text. However, there is a learning curve and a necessary period of familiarisation before a palaeographer would be able to use the interface with sufficient ease and generate transcriptions at the same speed as with direct access to the manuscript.

One of our participants in our user study was the palaeographer who eventually performed the transcription of the Great Parchment Book, and as a consequence of this preliminary study used our system for part of the transcription process.

Chapter 7

Global Flattening

In Chapter 4 we examined the effect of applying standard, general purpose surface parameterization algorithms to the parchment surfaces. The objective of mesh parametrization is to compute a map that flattens a 3D surface into the 2D plane while introducing as little distortion as possible. This is typically achieved by defining some geometric measure of distortion which the algorithm attempts to minimize in the mapping.

Computing a global, conformal or isometric parametrization may be sufficient for simpler deformations, but is not an adequate solution in general. The 3D geometry of the parchments already contains the deformation which we are trying to undo. The conformality and isometry metrics are computed from this geometry and so respecting these metrics would, in some sense, preserve the distortion while flattening the document.

In Chapter 6 we presented our interactive document viewer which circumvents this difficulty by approaching the problem from a different perspective. The approach successfully enables a user to read and explore a parchment folio while faithfully representing its current damaged state, however it is still preferable for many applications to have a single 2D representation of a folio. Interacting with 2D images is more familiar to the vast majority of potential users than interacting with 3D models, current archival practices are designed to handle images, not 3D models. Also, other tasks such as printing a folio, or displaying it in other 2D mediums such as a PDF or presentation slides, are far easier if a 2D image of the entire folio is available.

In this Chapter we present a method to estimate the global deformation

undergone by each folio and to obtain the restored, flat version of the folio by reversing the deformation.

7.1 Approach

Our global flattening approach takes as input a parchment folio digitised as a 3D triangle mesh using, for example, the digitisation pipeline proposed in Chapter 5. It is worth noting, however, that the flattening method described in this chapter is not dependent on that particular digitisation process. Letting $\mathcal{S} = \{\mathcal{V}, \mathcal{T}\}, \mathcal{V} \in \mathbb{R}^3$ represent the parchment surface, consisting of a set of vertices $\mathcal{V}$ and a set of faces $\mathcal{T}$, we denote by $f : \mathbb{R}^2 \rightarrow \mathcal{S}$ the deformation of the parchment. Our approach is to estimate f and invert it, thus restoring the original shape of the parchment.

Our method operates in two steps. We first estimate the Jacobian $\mathbf{J}_f$ of f, a per-triangle first-order approximation of the deformation, by analysing the texture of the document. We then use $\mathbf{J}_f$ to compute f^{-1}.

We estimate $\mathbf{J}_f$ based on a sparse set of constraints, described in Section 7.1.1, which we then interpolate over the entire document, as described in Sections 7.1.2 and 7.1.3. The entire pipeline is designed to provide interactive feedback and to enable a domain expert, such as a conservator or palaeographer, to repeatedly refine the estimate of $\mathbf{J}_f$ by defining additional constraints, and see their effect, in real-time.

As a final optional step we use an image-based technique to remove the shading and discolouration from the parchment, improving legibility while maintaining the character of the text.

7.1.1 Constraints

Our approach relies on the knowledge that, before being damaged, the text in the documents was written in equally spaced horizontal lines, with a uniform 'font size',and with vertical page margins. We, therefore, attempt to find an *orientation field* which captures the text direction at each point on the document surface, and a *scaling field* which captures the degree of shrinking or stretching of the text at each point.

The orientation field is represented by a 3D unit vector for each triangle in the mesh which sits in the plane of the triangle and runs parallel to the text, and the scaling field is represented by two scalar values per triangle which capture the scale of the text both parallel to and perpendicular to its orientation.

Line Constraints. The orientation field is estimated by tracing baselines through the text, and then interpolating their directions over the entire folio. Text baselines can found by rendering a portion of a parchment mesh into an image and then analysing its gradient using image processing techniques. Specifically, given an image I, a text baseline corresponds to a minimum in the vertical gradient, I_y, of the image. The gradient image can be computed by convolving I with a derivative of Gaussian kernel. We define for each pixel $\mathbf{p}_i$ in I a unary cost equal to the pixel value in I_y (scaled between 0 and 1), and for each pair of pixels $\mathbf{p}_i, \mathbf{p}_j$ in adjacent columns of I we define a binary cost equal to a constant step cost k if the rows of $\mathbf{p}_i$ and $\mathbf{p}_j$ differ, and ∞ if the rows differ by greater than some maximum step size $s_{\max}$. A minimum cost path through the image, corresponding to a text baseline, can be found using dynamic programming.

Line detection is complicated by the distortions, discolourations, faded text, and pronounced ascending/descending characters which are found in the parchments. These factors can cause the line tracing method to occasionally jump between different text lines. Therefore, we use a semi-automatic approach.

Once a user begins to manually trace a line on the flattened folio, the system continuously proposes a suggestion of the next section of the line, computed using the same dynamic programming approach described above. The user can either accept the suggestion or manually define the next point of the line, forcing the system to generate a new suggestion beginning at the manually defined point. This process is illustrated in Figure 7.1. Vertical lines can also be defined along the margins of the folio. However, these tend to be extremely faded or even not visible at all and must be marked manually.

Each triangle in the mesh which has a line constraint passing through it has its text orientation vector fixed to be parallel to that line constraint.

Attempts were made to use the orientation fields already described in Chapter 6. However these alone did not work sufficiently well. As we will

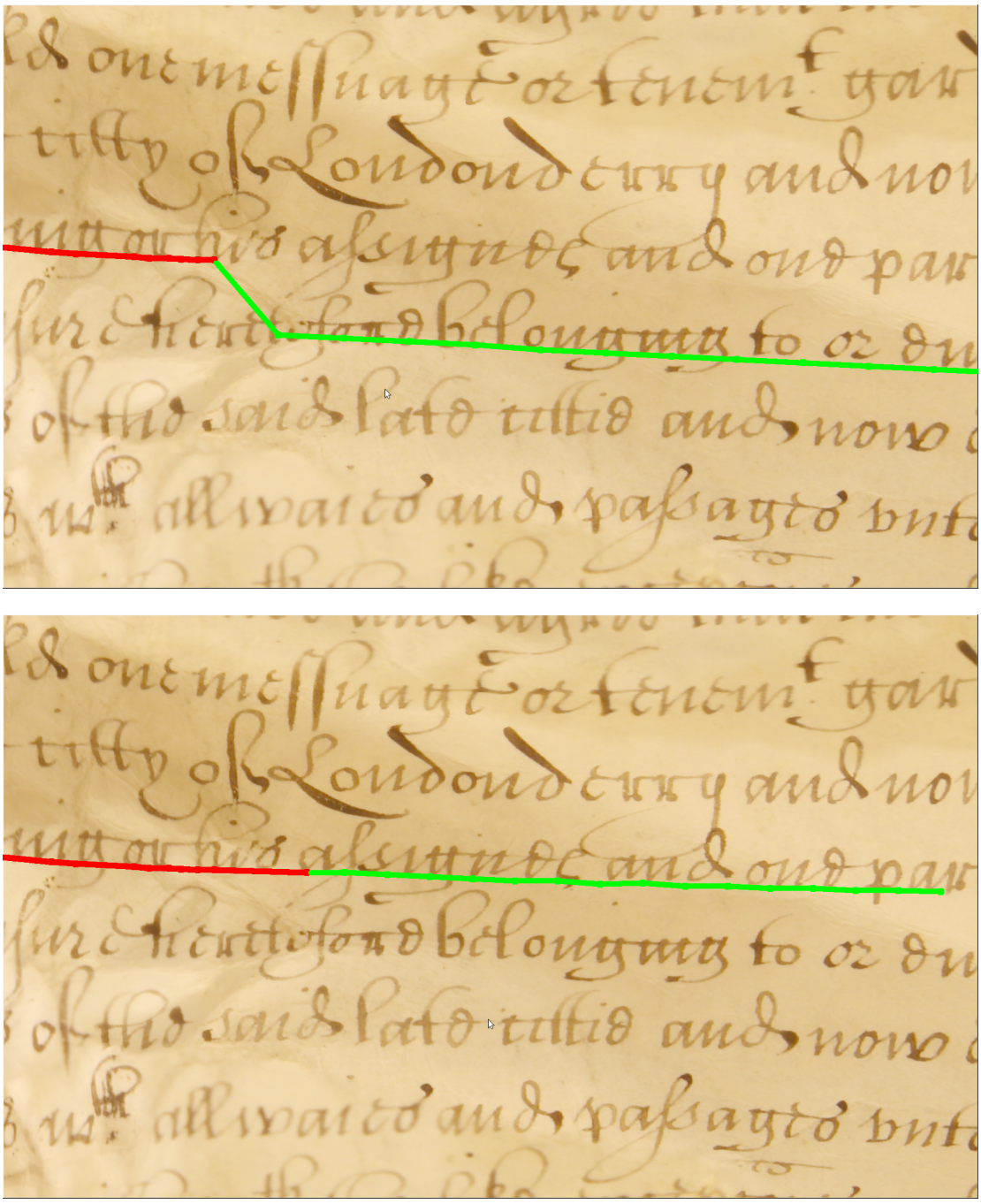

Figure 7.1: *Top*: The automatic line tracing (green) is misguided by the presence of ascenders and descenders. *Bottom*: After after a single user correction, the suggestion moves back to the correct baseline.

discuss in Section 7.1.4, the precise locations of text baselines is an important component of our global flattening method and this information isn't provided by Chapter 6's orientation field.

Scale Constraints. An initial estimate of the scaling field is automatically computed by detecting a sparse set of single characters (specifically, those without

ascenders or descenders) on the parchment surface using template-matching-based optical character recognition (OCR). The bounding box of these characters corresponds to the x-height (or corpus size) of the text. Since all of the text (excluding initials and marginalia) was originally written in approximately the same size, the relative sizes of characters in different regions of the document indicates the amount of shrinking or expansion that has occurred in that region. To detect characters, we apply a pattern-matching approach using templates of a small selection of characters. The templates are generated by manually extracting a number of examples of such characters from the parchment data set and aligning and blending them together. On a new parchment, characters are detected by automatically extracting roughly aligned patches of text at various locations on the parchment surface, using an approach similar to the local region flattening approach in Chapter 6. For each patch and character template, we compute the normalized cross correlation response image, identifying peaks as matches. To account for the scale variation across the parchment, the matching is performed with a range of scales for each template.

We find that instances of the letter *a* work particularly well as they typically have a more distinctive shape in the script of the parchments than other letters. Because they are very common, we can always detect a sufficient number in each document. The scale at a detected character location is defined as the distance between its top and bottom bounding edges. Figure 7.2 shows a sparse set of detected characters and a visualization of the isotropic scaling field computed by interpolating the scales from this set. We describe how this interpolation is computed in Section 7.1.3.

In regions with high anisotropic distortion, where the parchment has undergone significantly more shrinking in one direction than another, the OCR approach can fail to detect characters effectively, and the degree of anisotropy cannot be accurately detected. In these cases, we employ additional user input to specify the anisotropic scale by annotating the flattened folio with ellipses, such as that in Figure 7.3. The anisotropic scale is defined as the ratio between the lengths of the axes of the ellipses.

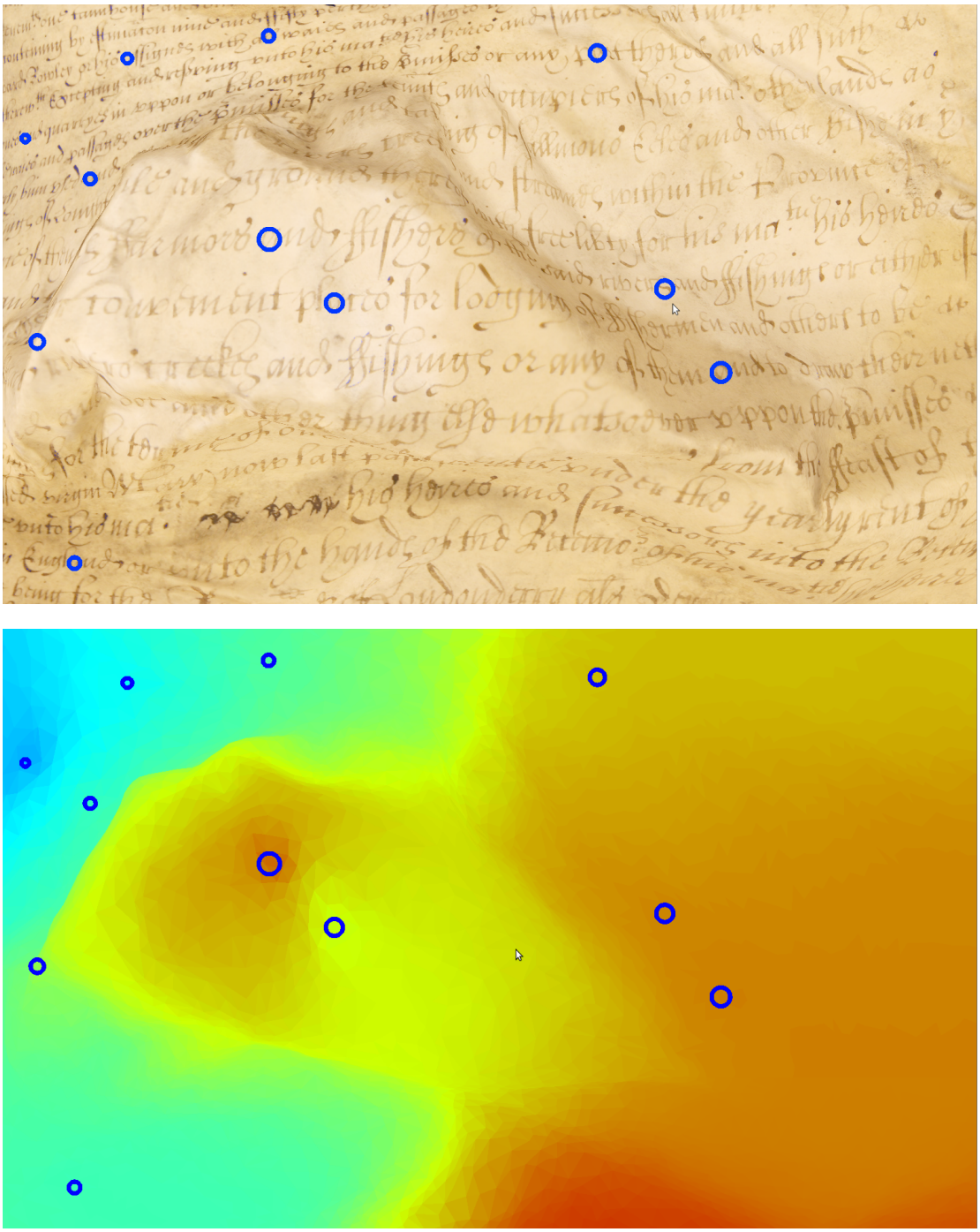

Figure 7.2: OCR detects characters in the texture. Their bounding boxes are used to estimate the uniform scaling.

7.1.2 Parameterizing the Deformation

For each triangle $t \in \mathcal{T}$ of the mesh, the Jacobian $\mathbf{J}_f(t)$ is a constant affine transform which can be represented as a $2x2$ matrix. In order to allow $\mathbf{J}_f$ to be easily interpolated over the entire surface, we represent it with three parameters

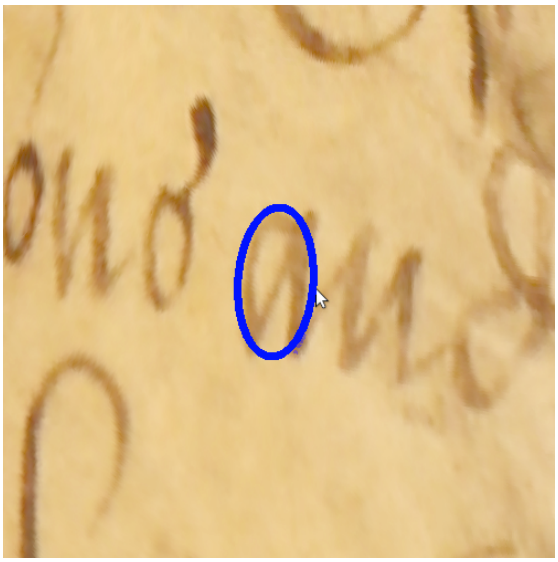

Figure 7.3: An anisotropic scale constraint provided by the user.

α, β, and γ:

$$\mathbf{J}_f(t) = \begin{pmatrix} \frac{\partial f_1}{\partial u} & \frac{\partial f_1}{\partial v} \\ \frac{\partial f_2}{\partial u} & \frac{\partial f_2}{\partial v} \end{pmatrix} = \begin{pmatrix} \alpha cos(\gamma) & \alpha sin(gamma) \\ \beta cos(\gamma + \frac{\pi}{2}) & \beta sin(\gamma + \frac{\pi}{2}) \end{pmatrix}, \tag{7.1}$$

where α and β are scaling factors which can be equal in the case of isotropic scaling of the text, or non-equal to represent anisotropic scaling, and γ is an orientation term representing the angle between the first row of $\mathbf{J}_f(t)$ and an arbitrarily chosen reference edge in the triangle. The scaling factors α and β are fixed for triangles which contain a scale constraint, and the orientation term γ is fixed for triangles which contain a line constraint. We then interpolate each of these parameters independently.

7.1.3 Interpolating the Deformation

We treat α and β as two separate scalar fields and interpolate them using a harmonic function to approximate the variation in scale over the mesh.We compute the vectors $\mathbf{h}_\alpha$ and $\mathbf{h}_\beta$, containing per-vertex values of α and β by solving the linear systems:

$$\begin{aligned} &\mathbf{L}\mathbf{h}_\alpha = \mathbf{0}, \text{ such that } \mathbf{h}_\alpha(i) = \mathbf{h}^0_\alpha(i) \text{ if vertex } i \text{ is constrainted.}, \\ &\mathbf{L}\mathbf{h}_\beta = \mathbf{0}, \text{ such that } \mathbf{h}_\beta(i) = \mathbf{h}^0_\beta(i) \text{ if vertex } i \text{ is constrainted.}, \end{aligned} \tag{7.2}$$

where $\mathbf{h}_\alpha^0$ and $\mathbf{h}_\beta^0$ are vectors containing the constrained values of α and β, and $\mathbf{L}$ is a discrete Laplacian matrix with elements defined as:

$$\mathbf{L}_{ij} = \sum_{j \in N(i)} \left(\frac{1}{2} \left(\mathbf{n}_i^T \mathbf{n}_j + 1 \right) \right)^\lambda, \tag{7.3}$$

for elements where $i \neq j$, and

$$\mathbf{L}_{ii} = -\sum_{j \neq i} \mathbf{L}_{ij}, \tag{7.4}$$

for elements on the diagonal.

The term for the non-diagonal elements $\mathbf{L}_{ij}$ is designed based on our observations on the parchment deformation. We see that sharp changes in scaling usually coincide with sharp changed in surface normal. We, therefore, use a term which penalises the diffusion of the scale parameters across sharp geometrical edges. This allows the observed scale change to be modelled with fewer constraints. The term λ is a scalar which controls the extent to which the difference in surface normal is penalised. We found that a value of $\lambda = 4$ works well, based on our experiments with various values.

To interpolate γ, we convert the orientations at constrained vertices into 3D vectors in the tangent plane of the surface, and then interpolate them using the method of Ray et al. [Ray et al., 2008] to generate a smooth vector field defined over $\mathcal{S}$ where all the vectors lie in the the tangent plane of $\mathcal{S}$. Their algorithm computes a direction vector for each face of mesh which interpolate a number of constrained direction vectors, defined on a subset of the faces, as smoothly as possible.

7.1.4 Inverting the Deformation

Following the interpolation, $\mathbf{J}_f$ is defined on each face on the mesh. Next, we show how $\mathbf{J}_f$ can be used to compute f^{-1}. Computing $\mathbf{J}_f$ given a set of constraints is very fast since each of the three interpolations (for α, β, and γ) can each be computed by solving a sparse linear system. There is no guarantee, however, that the resulting Jacobian $\mathbf{J}_f$ will be integrable, and that the function

f, whose Jacobian is $\mathbf{J}_f$, exists. We, therefore, look for an approximation of this f, or equivalently, we will modify $\mathbf{J}_f$ to make it integrable by solving a non-linear variational problem.

We use the inverse function theorem:

$$\mathbf{J}_{f^{-1}}(t) = \mathbf{J}_f^{-1}(t), \tag{7.5}$$

and compute f^{-1} as the function whose Jacobian matches $\mathbf{J}_{f^{-1}}$ in the least squares sense. We compute $f^{-1}(\mathcal{S})$ as the set of mesh coordinates $(\mathbf{u}, \mathbf{v})$ which solve the following Poisson problem:

$$f^{-1}(\mathcal{S}) = \underset{\mathbf{u},\mathbf{v}}{\operatorname{argmin}} \sum_{t \in \mathcal{S}} \left\| \begin{pmatrix} \nabla\, \mathbf{u}(t) \\ \nabla\, \mathbf{v}(t) \end{pmatrix} - \mathbf{J}_{f^{-1}}(t) \right\|_F^2, \tag{7.6}$$

In order to guarantee that the previously specified lines of text are perfectly horizontal when flattened, we introduce additional hard constraints for each pair of points $p_i, p_j \in \mathcal{S}$ in a given detected line:

$$f_u^{-1}(p_i) = f_u^{-1}(p_j), \tag{7.7}$$

Note that typically p_i will not correspond to a vertex of $\mathcal{S}$, but will be a point inside a triangle. In this case p_i is expressed as a linear combination of the three vertices of the triangle that contains it. A similar constraint is also used for the margins and other vertical lines, which constrain the points on those lines to be vertical:

$$f_v^{-1}(p_i) = f_v^{-1}(p_j), \tag{7.8}$$

Solving Equation 7.6 subject to the constraints in Equation 7.7 may not be sufficient if the parchment is heavily distorted. Regions of the mesh might flip their orientation when flattened, generating a result with overlapping triangles and hiding some portions of the text (see Figure 7.4). We introduce additional constraints to prevent this:

$$\det\left(\mathbf{J}_f^{-1}(t)\right) > 0, \quad \forall t \in \mathcal{S}, \tag{7.9}$$

Optimizing the energy in Equation 7.6 subject to Equation 7.7, Equation 7.8, and Equation 7.9 is a challenging problem. General purpose non-linear solvers could be used, but would be too slow for an interactive application. We instead use the solver proposed by Schüller et al. [Schüller et al., 2013], and described in Section 2.5, that is specifically designed to interactively minimize a deformation energy while keeping the determinant of the Jacobian positive, thus guaranteeing that no triangle flips are introduced. The effect of this positive Jacobian constraint is shown in Figure 7.4. In Figure 7.4 (Top) the positive Jacobian constraint is turned off and some of the triangles invert, causing the text to become illegible. In Figure 7.4 (Bottom), with the same set of textual constraints, the positive Jacobian constraint is turned on, and the words are legible.

7.1.5 Colour Normalization

Note that even after successful undistortion, the texture of the document still exhibits intensity and color variations which convey the false impression the document is still distorted and not flat. These variations are a combination of shading baked into the texture at the time of acquisition, and the genuine discoloration of the parchment which has taken place in the course of the damage. This discolouration is discussed in detail in Chapter 3. We find preserving these observed appearance variations a useful feature to study the rectified text in the context of the original damage, mitigating the risk of misinterpreting potential artifacts introduced by over-processing the content [Terras, 2011]. On a more general level, Bentkowska-Kafel [Bentkowska-Kafel, 2013] stress the importance of "Intellectual Transparency" in digital cultural heritage visualization, and the London Charter for the Computer-based Visualisation of Cultural Heritage [Denard, 2012] encourages the documentation of paradata, a detailed record of the process of generating the visualization. Preserving the original texture containing the observed appearance variations also preserves the veracity of the data, an aspect of vital importance for historians.

In many cases, however, a reader will prefer a cleaned-up color appearance in addition to the unwarped geometry. We hence optionally remove color variations by normalizing the parchment texture's appearance. This is achieved by

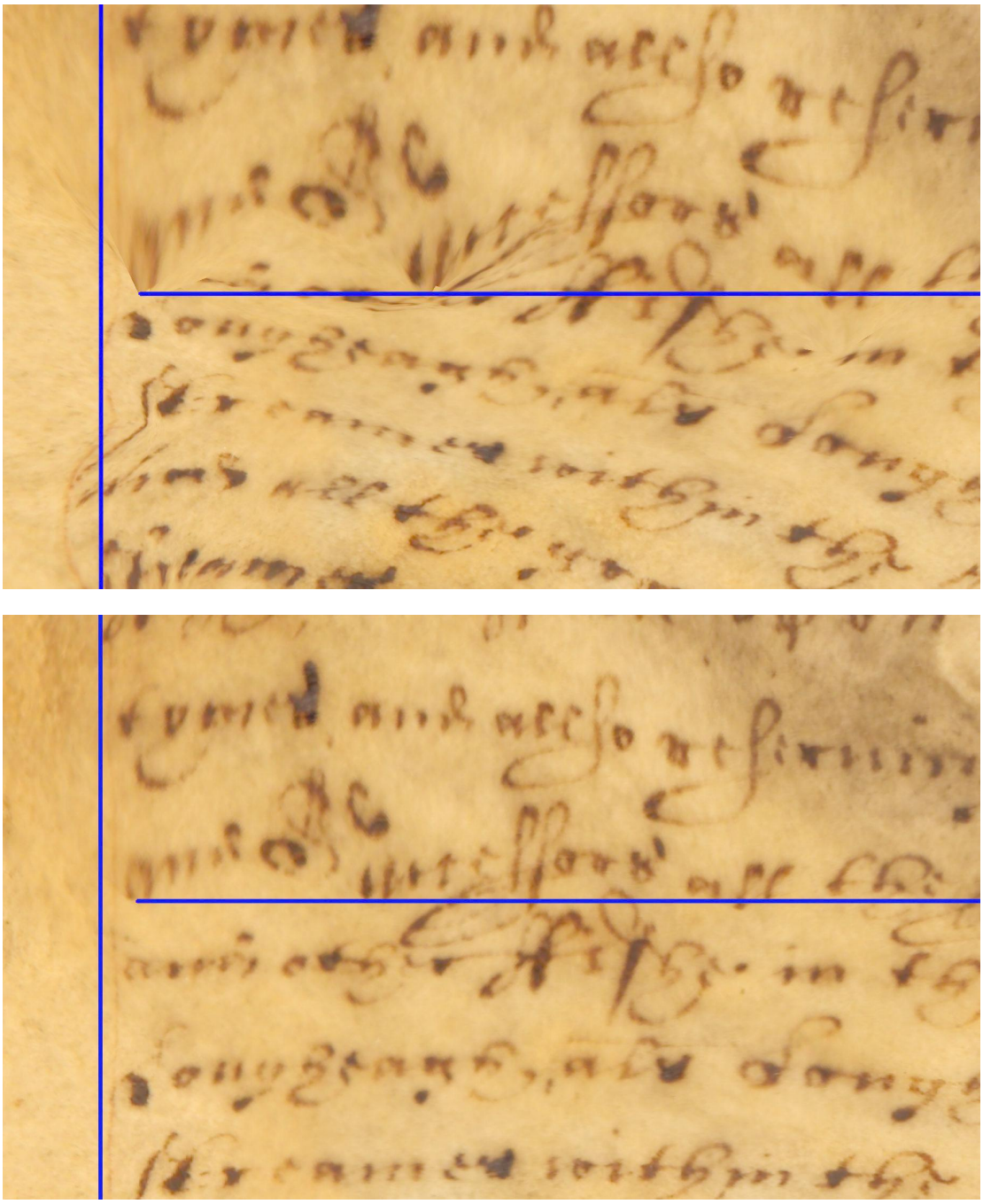

Figure 7.4: *Top*: A large region of the parchment is folded over, hiding many words of the script. *Bottom*: By constraining the determinant of the Jacobian to be positive, the fold-overs are eliminated, revealing the original script.

independently scaling each color channel by a spatially-varying factor, so that all ink-free regions of the parchment roughly match the same color. To determine this factor for a given surface point, we estimate its observed parchment color by sampling the 95-percentile of pixel intensities in a small, disk-shaped neighborhood. The percentile-based sampling introduces robustness against fine-scale

intensity variations (most importantly, those caused by ink), and the resulting smooth scaling field preserves subtle texture variations of the parchment despite the color normalization while removing coarse-scale color variations.

7.2 Results

Figure 7.5 shows the effect of different constraints in the virtual restoration process. By constraining only the boundary of the folio, a fairly good initial result can be obtained. However, this result still contains pronounced shrinking (visible along its left side), and the text bends around the creases. With additional line constraints, the bending is reduced, and the tear in the bottom left sealed, greatly improving readability. The text shrinking is only solved by introducing the isotropic constraints. The user-provided anisotropic constraints (highlighted in red) are used to fine tune the scaling in the left hand border of the folio.

Figures 7.6, 7.7, 7.8, and 7.9, show the results of applying our system on a variety of folios, with and without colour correction, rendered from a fronto-planar viewpoint. The results without colour correction show the rectification of the text, but the images still retain the illusion of a warped appearance due to the baked in shading, and also an impression of damage due to the discolouration of the parchment itself. By normalizing these variations away, the parchment looks more truly undistorted.

We successfully applied our algorithm on a number of folios of the Great Parchment Book with varying levels of damage. In all cases our algorithm requires only a relatively small number of constraints to restore the folios. Figures 7.10 and 7.11 show the results for these folios. The left column shows the 3D geometry of the damaged parchments, and the right column shows the flattened and colour-corrected result rendered from the same viewpoint. The blue lines show the constraints on each folio. The change in the constraint positions between the left and right columns clearly illustrates the effect of the un-distortion process, and how the text is rectified.

7.3 Discussion

In this Chapter we have presented an interactive method to virtually restore severely damaged and distorted documents. The method takes as input a textured 3D triangle mesh of the document, and produces as output a flattened, undistorted image. The entire pipeline of the method is shown in Figure 7.12.

We have shown how other parametrization approaches are unsuitable for this problem because are designed to maintain geometric surface metrics which know nothing about the distortion in the documents. Instead, we directly estimate the distortion using a sparse set of constraints based on text line and character features in the texture of the documents, and then invert this distortion.

We have adopted an interactive, semi-automatic approach to this problem to enable an expert user, such as a conservator or palaeographer, to efficiently guide the estimation of the distortion. Finally we optionally remove colour variations caused by shading in the acquisition and genuine discolourations on the parchment.

We have demonstrated our algorithm on a variety of folios from The Great Parchment Book. Staff at London Metropolitan Archives are currently in the process of being trained how to use the software we have produced, with which they plan to flatten the remaining folios of the book. In Chapter 8 we discuss the future plans and outlook of the project in more detail.

Our method allows the entire textual content of a folio to captured in a single image, which is impossible to using the standard archival method of taking a single fronto-parallel photograph. This representation as a 2D image allows the entire text to be disseminated in various was, such as displayed on websites or printed in books. This ease of dissemination could also have further implications in terms of transcription. The idea of crowdsourcing the transcription of documents has been explored in other contexts [Causer and Terras, 2014] with documents that are easy to visualise as images. Since our method allows the text of heavily distorted documents to be visualised in a single image and viewed remotely without the need for access to the original manuscript, it opens up new possibilities such as crowdsourced transcription. to be easily visualised (b

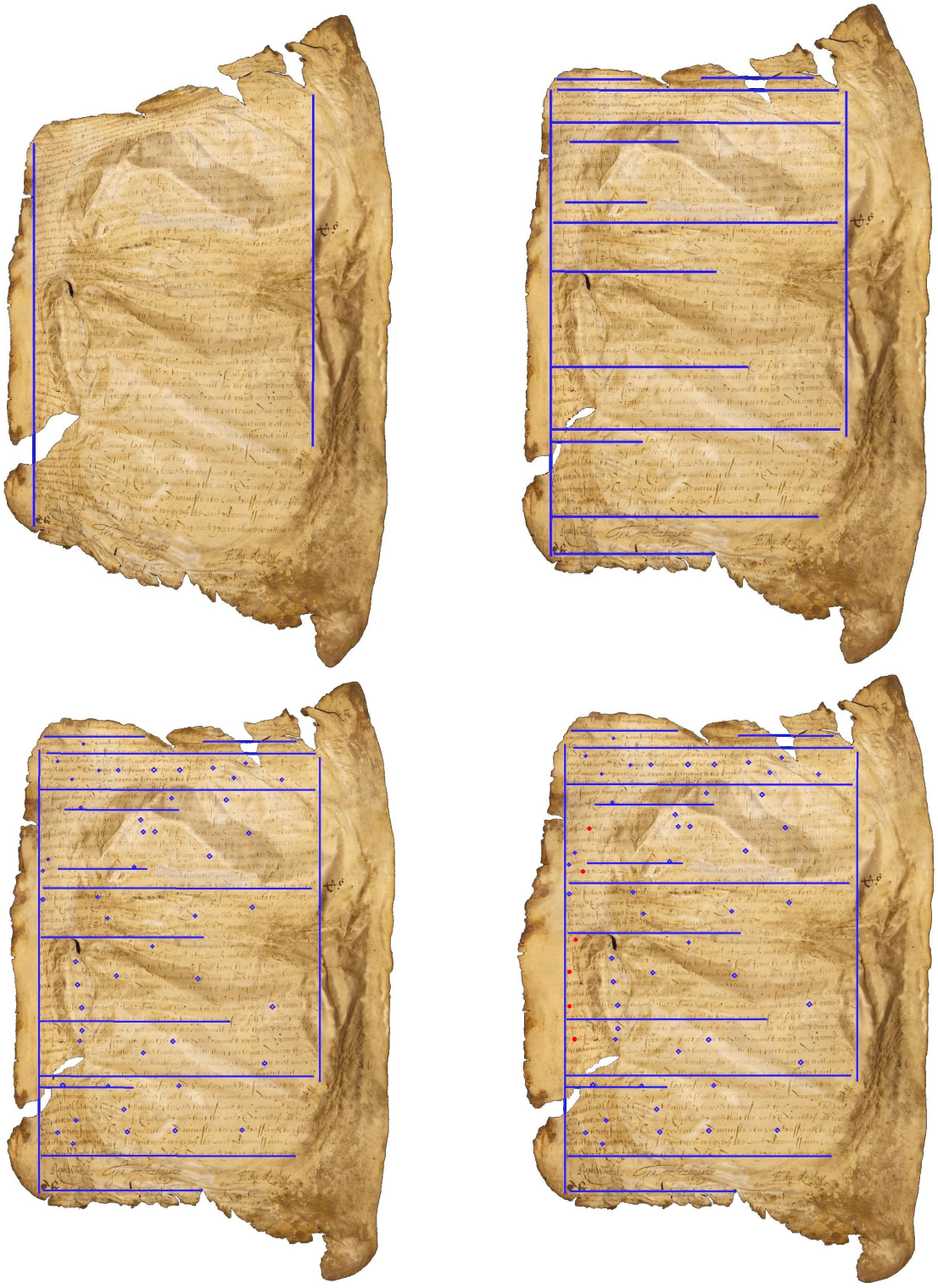

Figure 7.5: Effect of different constraints in the virtual restoration process. *Top Left:* Border only. *Top Right:* Border plus additional text lines. *Bottom Left:* Line constraints plus isotropic scaling. *Bottom Right:* Line constraints plus isotropic and anisotropic (red) scaling.

Figure 7.6: Folio A1a of the Great Parchment Book restored by our algorithm, shown with and without removal of shading and discolouration, and with a detail image showing a close-up view of a region of text.

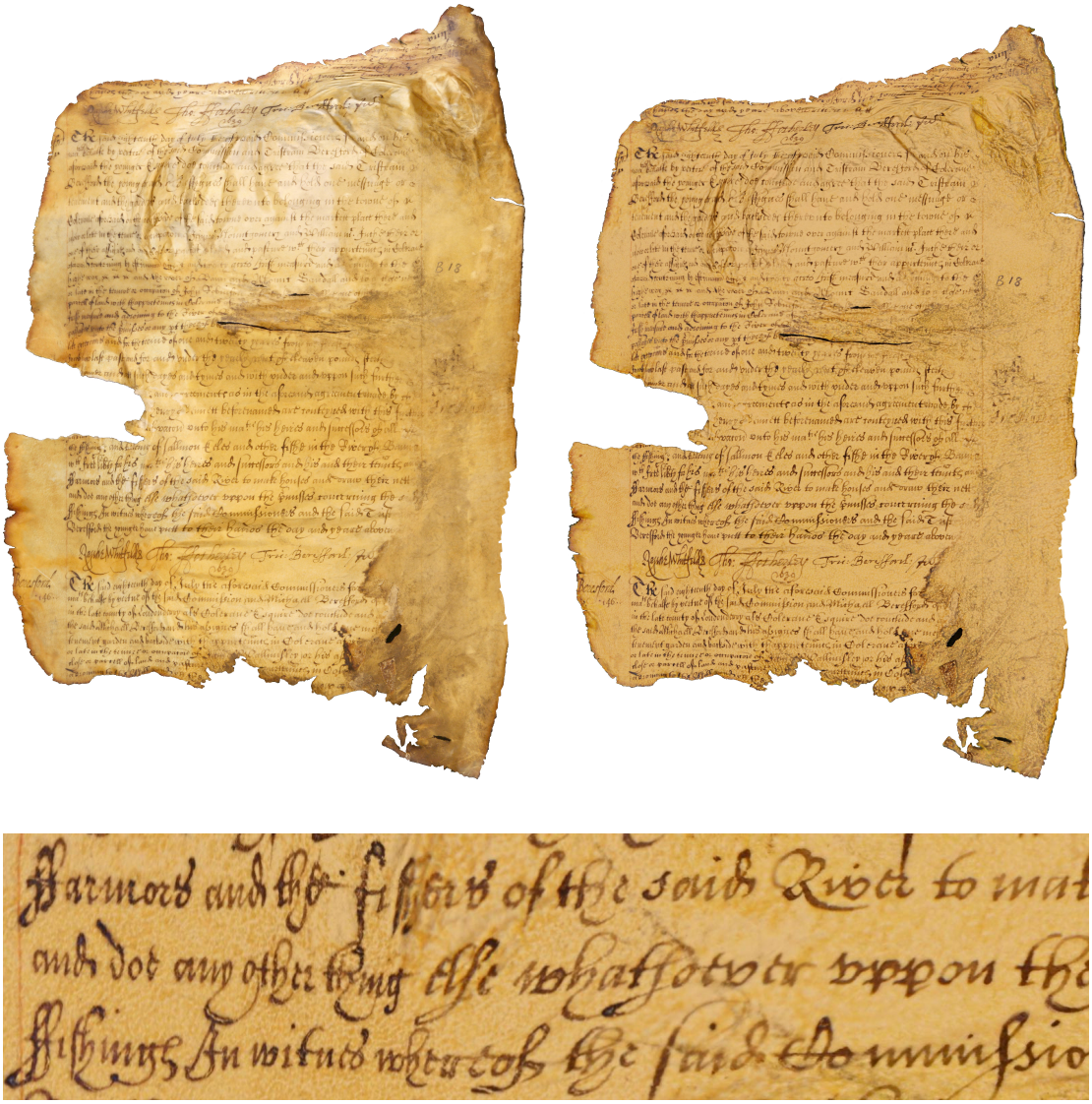

Figure 7.7: Folio B17b of the Great Parchment Book restored by our algorithm, shown with and without removal of shading and discolouration, and with a detail image showing a close-up view of a region of text.

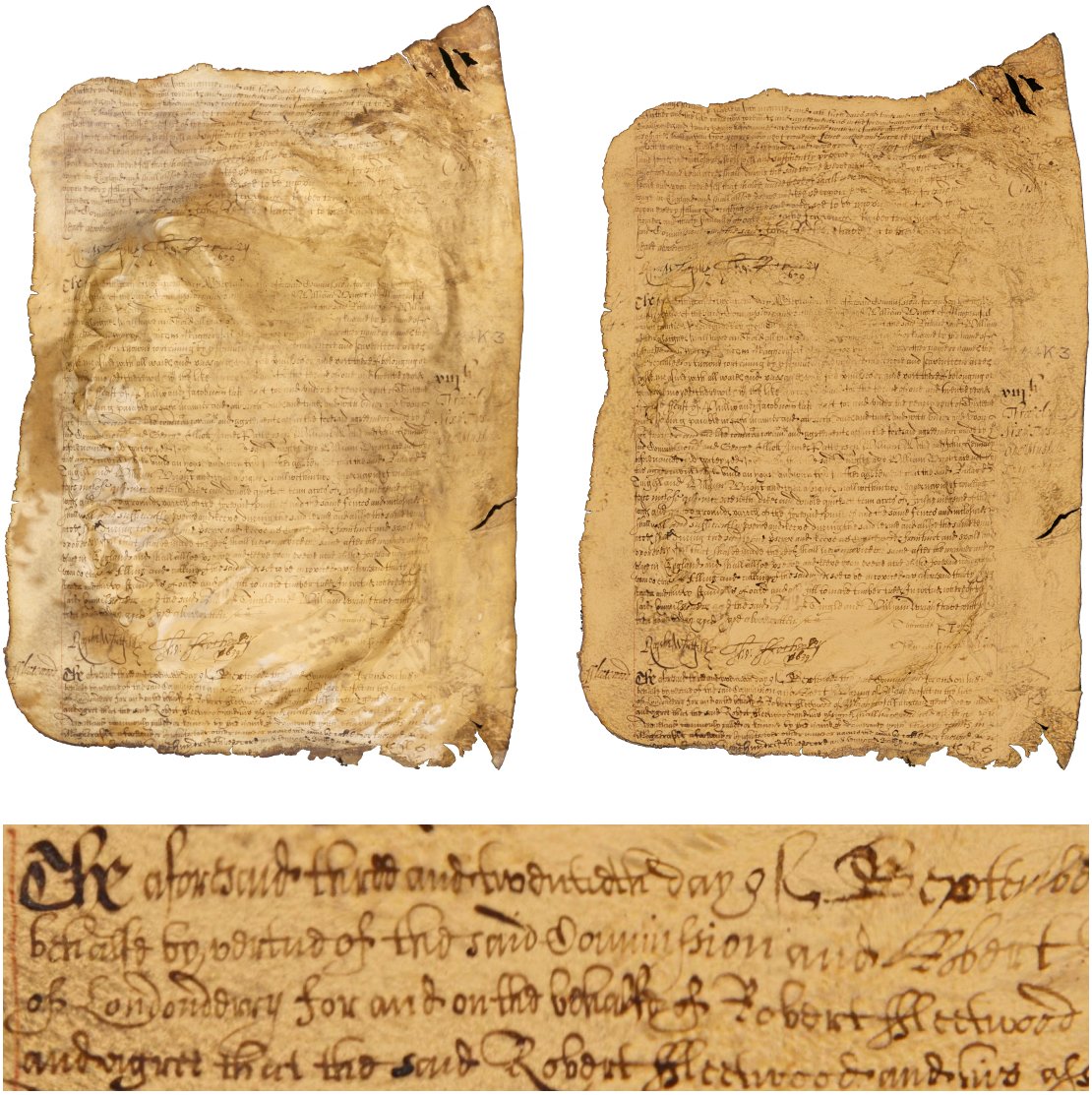

Figure 7.8: Folio K3a of the Great Parchment Book restored by our algorithm, shown with and without removal of shading and discolouration and with a detail image showing a close-up view of a region of text.

Figure 7.9: Folio N3a of the Great Parchment Book restored by our algorithm, shown with and without removal of shading and discolouration and with a detail image showing a close-up view of a region of text.

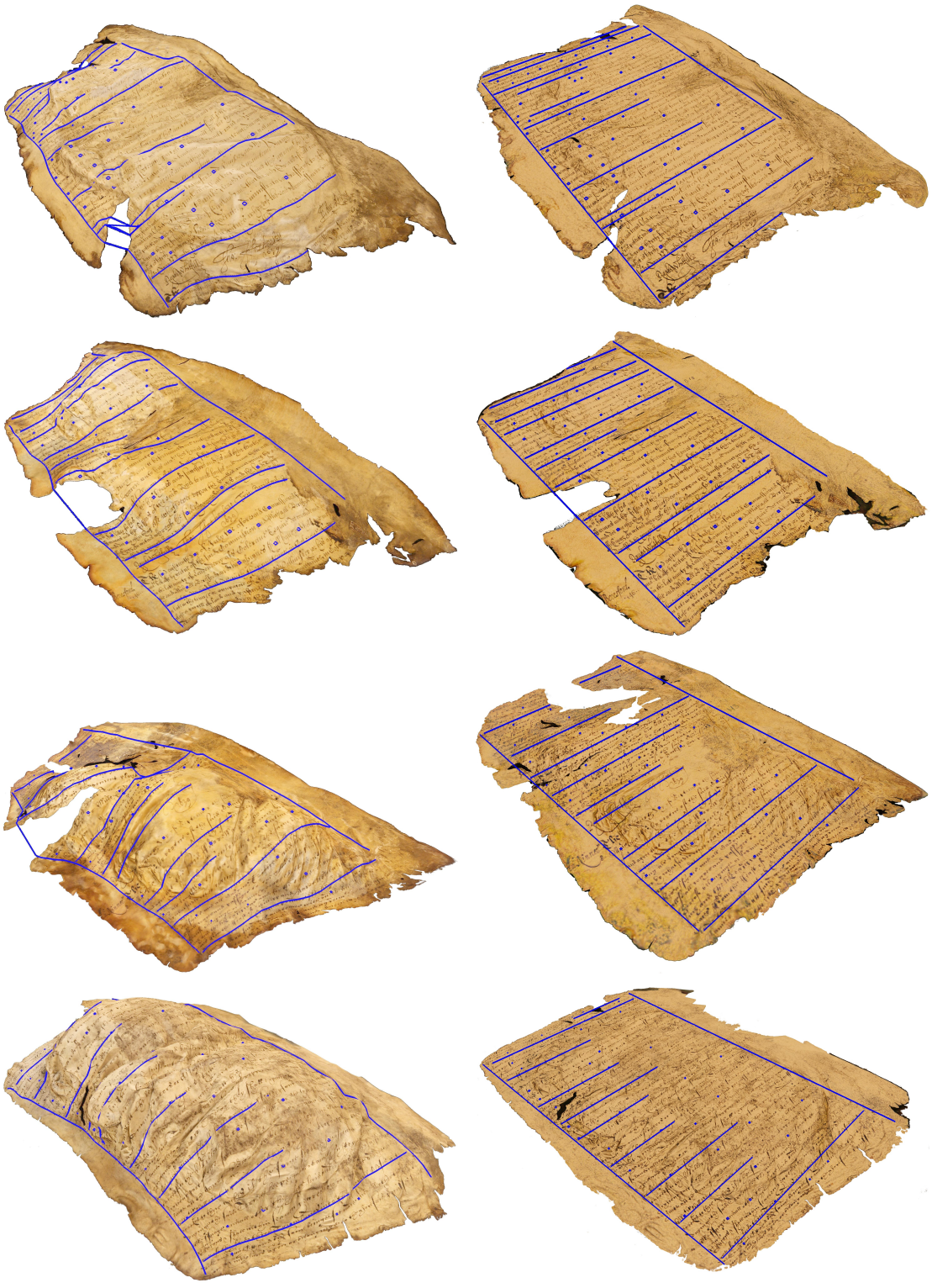

Figure 7.10: *Left:* The distorted parchments are annotated with the constraints shown in blue. *Right:* The documents are virtually restored, flattening the folios and removing the distortion.

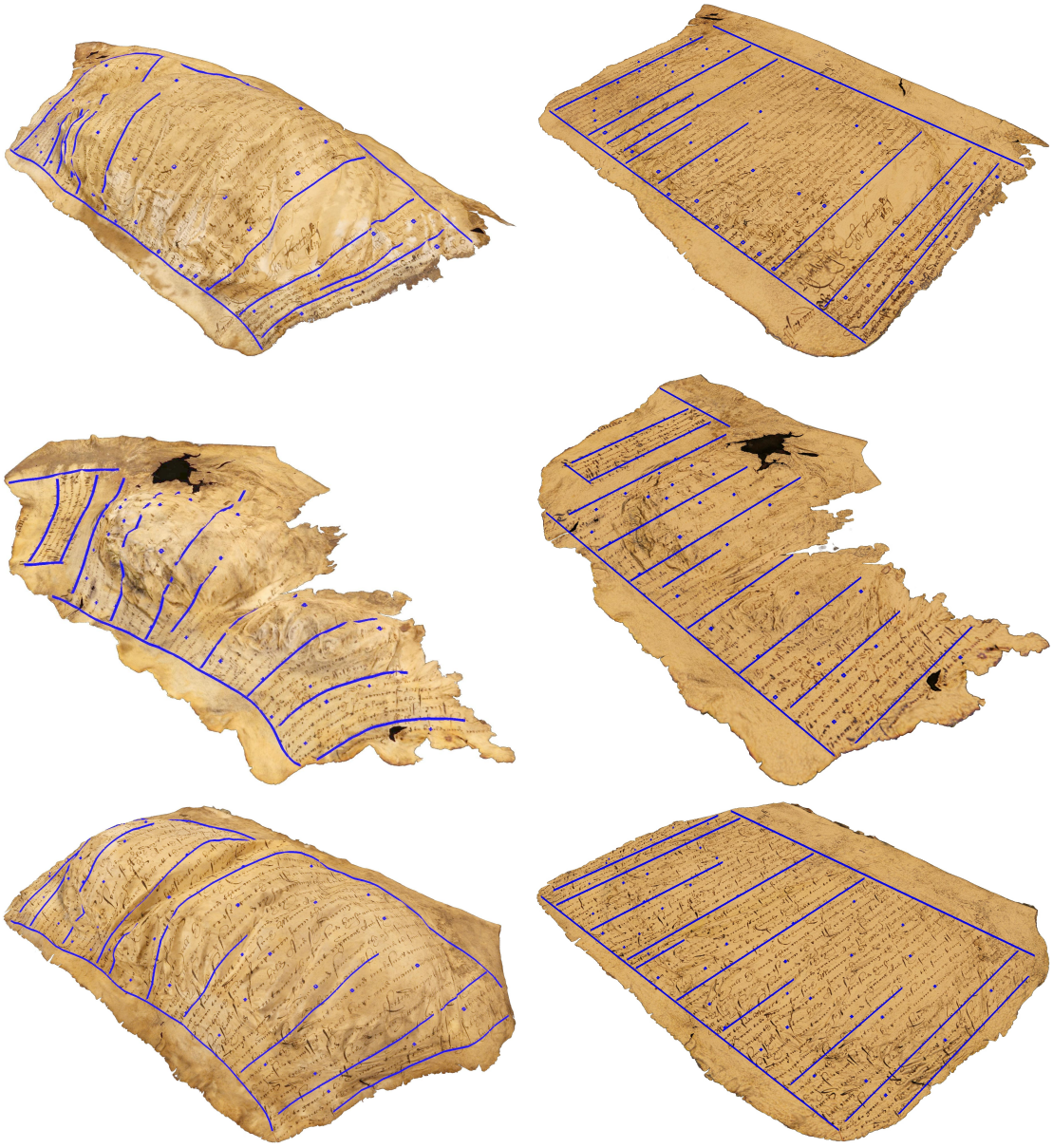

Figure 7.11: *Left:* The distorted parchments are annotated with the constraints shown in blue. *Right:* The documents are virtually restored, flattening the folios and removing the distortion.

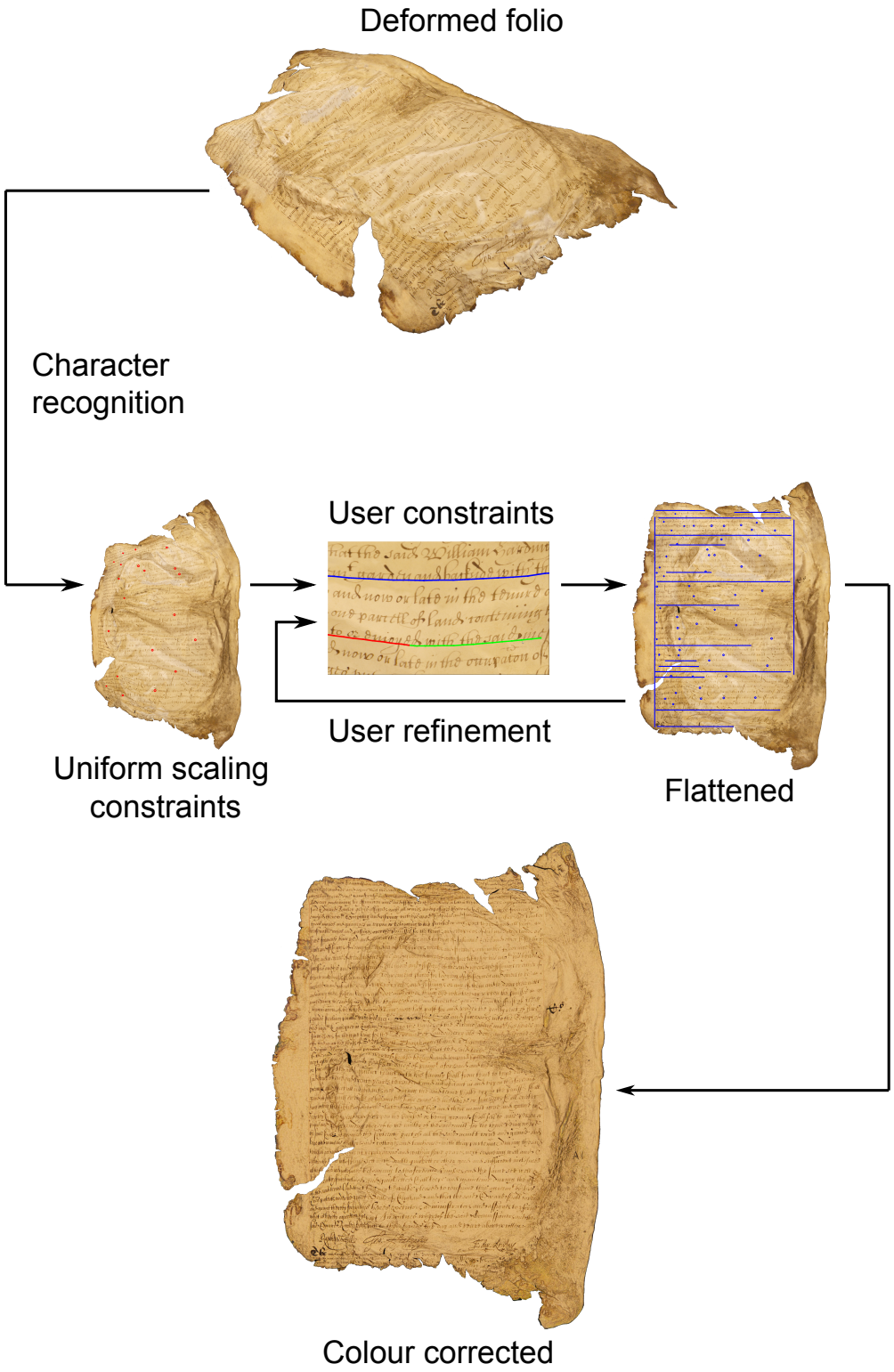

Figure 7.12: All steps of our algorithm. *Top*: The original distorted folio. *Middle*: The folio is flattened using the Jacobian estimated from the OCR analysis. An interactive refinement of the constraints improves the restoration, and is iterated until a satisfactory result is achieved. *Bottom*: Finally, we remove the intensity and color variations from the texture.

Chapter 8

Conclusions and Future Work

This chapter provides a summary of the thesis and a discussion of the outlook of the project. In this thesis we have presented a pipeline of techniques for digitising, visualising, and digitally restoring historical documents written on parchment, which have been damaged to the point that they have adopted a 3D shape and cannot be conserved in the traditional manner. Our pipeline allows such documents to be digitally catalogued, studied, and disseminated in a variety of ways.

One such document, which has been the focus of our work, is The Great Parchment Book, a 17th century document containing a survey of estates of the Northern Irish county of Londonderry. It is an extremely valuable resource to historians, but was damaged in a fire in 1786 and has been kept in storage ever since. It is currently in the possession of London Metropolitan Archives, with whom we collaborated on this project.

As its name suggests, the book is written on pages made of parchment. In Sections 2.1 and 3.2, we discussed how the organic nature and irregular internal structure of parchment can cause it to deform in very unpredictable ways when exposed to high temperatures and/or moisture, adopting a three dimensional shape. In cases of extreme damage, the methods used by document conservators to physically restore and conserve parchment cannot undo such damage without the risk of doing further damage to the parchment. As an alternative solution, we developed a digital restoration pipeline, consisting of a lightweight digitisation method to generate a digital surrogate of a folio, and two complementary flattening systems aimed at improving the legibility of the

surrogate.

Our pipeline begins with a 3D reconstruction method, detailed in Chapter 5, to digitise the parchment folios, in a way that fully captures their deformed shape. Each folio is imaged using a hand-held digital-SLR camera and the resulting image set, typically consisting of between 40 and 60 images, is used to generate a 3D reconstruction, in the form of a textured triangle mesh, of each parchment using a structure-from-motion and multi-view-stereo-based approach. Since the shape of different folios can differ significantly, this approach allows the imaging process to be easily adapted to the specific shape of a given parchment. Using this method, we have digitised all of the pages of The Great Parchment Book. Along with the digitisation pipeline, in Section 5.4 we also show how to compute a measure of quality for a folio digitisation. Since standardised guidelines for imaging of 2D documents are given in terms DPI [Federal Agencies Digitization Initiative, 2010, Library of Congress, 2007], we compute our quality measure by analysing the distribution of the DPI (sampling frequency) at which the surface of the folio is imaged. This allows the 3D digitisations to be compared against these standards in a meaningful way.

While the 3D digitisation allows the true physical state of a document to be accurately archived, flattening the 3D surrogate into 2D can help to improve the legibility of the digitisations, and allow archivists to deal with them in the same way as they would a 2D digitisation. Previous work has already made the analogy between document flattening and surface parametrization [Brown and Seales, 2000, Brown et al., 2007, Samko et al., 2014]. However, as we discuss in detail in Chapter 4, the types of deformation present in damaged parchment can cause standard surface parametrization algorithms to fail. We proposed two approaches to deal with these complex deformations.

Firstly, exploiting the fact that a reader will only ever inspect a small area of the folio at a given time, we proposed a method for performing local undistortion of the parchments inside an interactive viewer application. The application makes an analogy between document flattening and map-making. As an alternative to flattening the entire document, the application, described in Chapter 6, allows a user to browse a parchment folio along and perpendicular

to the lines of text, much like browsing a digital globe along lines of latitude and longitude. As the user browses the document surface, the application generates local maps, in real-time, of the area of parchment currently under inspection by applying least-squares conformal-mapping [Lévy et al., 2002] to the region of the surface currently on-screen. It also maintains the provenance of the image data by allowing the user to refer back to the original image set of the folio. By clicking on a point on the 3D reconstruction, the user can have the system select the image from the original data set which sees that point best, and display the corresponding region of that image, scaled and rotated to rectify it with the on-screen view of the reconstruction. This can help the user to resolve ambiguities in the reconstruction and to decide whether what they are seeing is accurate.

Secondly, we proposed a method which can generate a global map of a document by directly estimating the deformation which has transformed the parchment into its current shape by using the knowledge that the text was originally written in straight lines and in a roughly uniform script size. We detect the variation in text orientation and size using a combination of automatically-detected and user-provided constraints, and use this information to estimate the deformation. In Chapter 7 we detail how these constraints are obtained, how they are used to estimate the deformation, and then show how this deformation can be inverted by posing the problem as a Poisson mesh deformation. We solve this Poisson problem in a way that guarantees local injectivity [Schüller et al., 2013], ensuring that no fold-overs are introduced in the result, to generate a globally flattened and undistorted image of each folio. We also show how these images can optionally be colour corrected to remove the shading cues baked into the reconstruction texture, and the discolourations in the parchment itself, to further improve legibility and give a more complete impression that the parchment has been restored.

8.1 Outlook and Future Work

The systems we have developed have been very well received by London Metropolitan Archives, as well the the larger archival community. The feedback we have received suggests that our methods would not replace existing archival

approaches outright, but would rather augment existing workflows and provide new capabilities for the institutions employing them.

As of the time of writing of this thesis, staff at London Metropolitan Archives are being trained in the use of our flattening software so that they can apply it to all of the remaining folios of The Great Parchment Book. Some of these flattened images, along with transcriptions, photographs, and other information, can be found at `http://www.greatparchmentbook.org/`. More flattened images will be made available at this URL as they are generated.

LMA have also been investigating potential future directions by experimenting with other interesting documents in their collections. Their feedback so far indicates that our methods are of sufficient usability that they could be adopted as a part of their workflow. They are also sufficiently pleased with the results of the project and the scope of other possible use cases that (as of the time of writing of this thesis) they are discussing the possibility of providing our software and techniques as a service for other heritage institutions who would find them useful.

Although the project was largely successful, our work has a number of limitations which point towards possible areas of future work.

Geometry vs. texture resolution The quality measure proposed in Chapter 5 addresses the issue of texture quality but not the issue of geometry quality, which is also a crucial factor in assessing overall reconstruction quality (in fact our method assumes accurate geometry in order to compute a meaningful texture quality measure). Our method could be augmented to take geometry quality into account. One possible approach would be to consider the initial sparse point reconstruction generated by VisualSFM during the reconstruction process. The points in this reconstruction are generated by triangulating image keypoints which were detected across multiple images. The density and distribution of these points tells us something about how well the geometry of different areas of the parchment has been captured. A high point density means that the reconstruction algorithm has managed to detect many features on the parchment surface successfully, suggesting a high reconstruction quality, whereas a low point density indicates that not many features have been detected, and thus the

local surface geometry has not been captured at a high resolution.

Presenting this information to archivists, however, might be more challenging than than the texture quality information. The archival community has a well established standards and a ready-made language for discussing texture resolution, but no such standards or language exists for geometry resolution. Therefore, how these statistics could be presented in a meaningful way to archivists would also be an important consideration.

Imaging limitations / processes to automation of the techniques adopted The imaging process we have proposed in Chapter 5 is flexible and does not require specialist hardware. However, it becomes very time-consuming when capturing a large number of folios and requires a human operator not only to be present but to be physically active throughout the capture process. Alternative approaches such as camera domes [Hawkins et al., 2001] or turntable-based approaches [Fitzgibbon et al., 1998] could speed up the process and reduce the workload of the operator, but would also lose some flexibility since they fix the camera positions and cannot, therefore, be adapted to different parchment shapes. A more sophisticated approach could involve a method known as next-best-view planning [Maver and Bajcsy, 1991, Scott et al., 2003], where an algorithm is used to automatically select optimal camera poses during the imaging process and generate a complete reconstruction with no holes. By using such an algorithm to control a camera mounted on a robotic arm, the capture process could be made largely automatic, significantly reducing operator workload while retaining the ability to adapt the capture process to different folio shapes.

It is possible that, in any given digitisation project, the variability in the shape of the folios might be quite small, removing the need to adapt the camera poses. In such cases a dome or turntable approach might be ideal. A one size fits all approach is not possible, especially when the different budgets available to different projects must also be taken into account. Instead, the benefits of each method must be weighed on a case-by-case basis against the requirements and budget of a given digitisation project.

Appendices

Appendix A

Glossary

Here we present a glossary of various terms used in this thesis. Since the thesis is of potential interest to readers from a wide variety of backgrounds, the aim of this glossary is to explain the meaning of technical terms to those with a humanities background.

3D reconstruction

The process of capturing a virtual representation of the shape and appearance of a real-world object. A wide variety of methods exist for doing this, which are discussed in Chapter 2 Section 4.

Anisotropic

Directionally dependent. For example the effective DPI measurements in Chapter 5 have a degree of anisotropy, and the scale constraints in Chapter 7 can be anisotropic if the parchment has shrunk more in one direction than another.

Camera calibration

The pinhole camera model is a simple model for describing the geometric properties of the camera used to take an image. It consists of intrinsic parameters, such as the focal length and principal point of the camera, and extrinsic parameters such as rotation and camera centre which describe the 3D position of the camera in the world. Camera calibration refers to the process of estimating these parameters.

Cross field

Very similar to a vector field. Given a space (which could be an image, a

3D surface, etc), a scalar field associates a cross (consisting of two vectors) to every point in that space.

Dynamic programming

A programming approach where a complex problem is broken down into a sequence of smaller overlapping subproblems (the result of one subproblem depends on the results of previous subproblems).

Energy minimisation (mathematical optimization)

The problem of choosing a best element from a set of elements, based on some criteria. Typically this involves describing the criteria as a mathematical function which takes a number of input values and returns a single output value, or energy, representing the goodness of those values (usually a small energy correspond to good input values), and then finding the values which minimise the energy.

Euler integration

A simple first order numerical method for solving ordinary differential equations subject to an initial value.

Fundamental matrix

A 3×3 matrix describing the geometric relationship between a stereo image pair.

Isometric deformation/mapping/embedding

A deformation of a surface in which distances (along the surface) between points on the surface are preserved. A real-world example is folding a piece of paper.

Linear system

A collection of linear equations involving the same variables. To solve the system, a value must be found for each variable such that all of the equations are satisfied. In some cases, a perfect solution may not exist, and so an approximate solution can be found which minimizes the error in the equations. A linear system is described as sparse if each equation in

the collection only includes a small subset of the overall set of variables. Computers can typically solve sparse linear systems much faster than a dense (non-sparse) system.

Mass-spring simulation

A method for simulating the movement of a deformable surface (e.g. parchment) by modelling it as a collection of small masses connected by springs, and acted upon by external forces such as gravity. The movement of the surface is simulated by computing the movement of the masses according to the laws of physics.

Multidimensional Scaling

The problem of embedding objects in some high-dimensional space into a lower-dimensional space such that the distances between the objects are preserved as well as possible.

Normalized Cross Correlation

In image processing applications, this is a measure of similarity between two images.

Poisson problem

A problem with an equation of the form $\nabla^2\phi = f$. Problems of this type appear in many situations in computer graphics and computer vision.

Runge-Kutta method (RK4)

A numerical method for solving ordinary differential equations subject to an initial value. It is a fourth-order method and is typically more stable than Euler integration.

Scalar field

Given a space (which could be an image, a 3D surface, etc), a scalar field associates a scalar value to every point in that space. The heatmaps in Chapter 5, for examples, are scalar fields on the surface of a parchment.

Segmentation

The process of dividing an image into multiple segments. A common type

of segmentation is a binary segmentation in which the image is divided into foreground and background (which correspond naturally to ink and non-ink in the case of a document image).

Singular Value Decomposition (SVD)

A method for factorizing a matrix M into three other matrices $M = U\Sigma V$, each of which have a well-defined form and meaning. The SVD has many applications in mathematics and computer science.

Surface parameterization

The problem of mapping a three-dimensional surface into two dimensions. This has many uses in computer graphics and various algorithms have been developed to solve this problem.

Triangle mesh

A set of triangles which share common vertices. This is a commonly used way of representing a three dimensional surface.

Vector field

Given a space (which could be an image, a 3D surface, etc), a scalar field associates a vector to every point in that space.

Appendix B

List of Publications

Content-Aware Surface Parameterization for Interactive Restoration of Historical Documents

Kazim Pal, Christian Schller, Daniele Panozzo, Olga Sorkine-Hornung, Tim Weyrich

Computer Graphics Forum (Proceedings of Eurographics), 33(2), pages 401-409, 2014

3D Reconstruction For Damaged Documents: Imaging of The Great Parchment Book

Kazim Pal, Melissa Terras, Tim Weyrich

In Proceedings of 2nd International Workshop on Historical Document Imaging and Processing, pages 1421, Washington DC, 24 August, 2013

Interactive Exploration and Flattening of Deformed Historical Documents

Kazim Pal, Melissa Terras, Tim Weyrich

Computer Graphics Forum (Proceedings of Eurographics), 32(2), pages 327-334, 2013

Geometric Analysis in Cultural Heritage

Ruggero Pintus, Kazim Pal, Ying Yang, Tim Weyrich, Enrico Gobbetti, Holly Rushmeier

In Proceedings of Eurographics Workshops on Graphics and Cultural Heritage, State-of-the-Art Report (STAR), pages 117, Darmstadt, Germany, Oct 68, 2014

Heritage Imaging at UCL

Alejandro Giacometti, Adam Gibson, Mona Hess, John Hindmarch, Lindsay Macdonald, Kazim Pal, Stuart Robson, Melissa Terras, Tim Weyrich

In Proceedings of Annual Conference of Association for Historical and Fine Art Photography (AHFAP), London, UK, 27 November 2014

The Great Parchment Book

Nicola Avery, Alberto Campagnolo, Caroline De Stefani, Kazim Pal, Matthew Payne, Philippa Smith, Rachael Smither, Ann Marie Stewart, Emma Stewart, Patricia Stewart, Melissa Terras, Laurence Ward, Tim Weyrich, Elizabeth Yamada
Poster presentation at Digital Humanities 2013, University of Nebraska, Lincoln. July 2013

Bibliography

[Ahmadabadian et al., 2014] Ahmadabadian, A. H., Robson, S., Boehm, J., and Shortis, M. (2014). Stereo-imaging network design for precise and dense 3d reconstruction. *The Photogrammetric Record*, 29(147):317–336.

[Arnold, 2008] Arnold, D. (2008). Digital artefacts: possibilities and purpose. In Greengrass, M. and Hughes, L., editors, *The Virtual Representation of the Past*. Ashgate.

[Arya et al., 1998] Arya, S., Mount, D. M., Netanyahu, N. S., Silverman, R., and Wu, A. Y. (1998). An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *Journal of the ACM*, 45:891–923.

[Barber et al., 2006] Barber, D. M., Dallas, R. W. A., and Mills, J. P. (2006). Laser scanning for architectural conservation. *Journal of Architectural Conservation*, 12(1):35–52.

[Bentkowska-Kafel, 2013] Bentkowska-Kafel, A. (2013). I bought a piece of roman furniture on the internet. It's quite good but low on polygons. *Digital Visualization of Cultural Heritage and its Scholarly Value in Art History, Visual Resources. An International Journal of Documentation, Special Issue on Digital Art History*, 29(1):38–46.

[Bertozzi et al., 2007] Bertozzi, A. L., Esedoglu, S., and Gillette, A. (2007). Inpainting of binary images using the cahn–hilliard equation. *IEEE Transactions on Image Processing*, 16(1):285–291.

[Besl, 1988] Besl, P. J. (1988). Active optical range imaging sensors. In Sanz, J. L. C., editor, *Advances in Machine Vision*, pages 1–63. Springer-Verlag New York, Inc.

[Besl and McKay, 1992] Besl, P. J. and McKay, N. D. (1992). A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14:239–256.

[Bloomenthal, 1994] Bloomenthal, J. (1994). *An Implicit Surface Polygonizer*, pages 324–349. Academic Press.

[Bommes et al., 2009] Bommes, D., Zimmer, H., and Kobbelt, L. (2009). Mixed-integer quadrangulation. *ACM Transactions on Graphics*, 28(3):77–87.

[Bronstein et al., 2006] Bronstein, M. M., Bronstein, A. M., Kimmel, R., and Yavneh, I. (2006). Multigrid multidimensional scaling. *Numerical Linear Algebra with Applications*, 13(2-3):149–171.

[Brown and Seales, 2000] Brown, M. S. and Seales, W. B. (2000). Beyond 2d images: Effective 3d imaging for library materials. In *Proceedings of the 5th ACM Conference on Digital libraries*, pages 27–36. ACM.

[Brown et al., 2007] Brown, M. S., Sun, M., Yang, R., Yun, L., and Seales, W. B. (2007). Restoring 2d content from distorted documents. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(11):1904–1916.

[Buehler et al., 2001] Buehler, C., Bosse, M., McMillan, L., Gortler, S., and Cohen, M. (2001). Unstructured lumigraph rendering. *Proceedings of the 28th annual conference on Computer graphics and interactive techniques (SIGGRAPH 2001)*, pages 425–432.

[Causer and Terras, 2014] Causer, T. and Terras, M. (2014). Crowdsourcing bentham: Beyond the traditional boundaries of academic history. *International Journal of Humanities and Arts Computing*, 8(1):46–64.

[Chan and Vese, 2001] Chan, T. F. and Vese, L. A. (2001). Active contours without edges. *IEEE Transactions on Image Processing*, 10(2):266–277.

[Clarkson, 1987] Clarkson, C. (1987). The preservation and display of single parchment leaves and fragments. In Petherbridge, G., editor, *Conservation of Library and Archive Materials and the Graphic Arts*, pages 201—209. Butterworth.

[Cohen-Steiner and Morvan, 2003] Cohen-Steiner, D. and Morvan, J.-M. (2003). Restricted delaunay triangulations and normal cycle. In *Proceedings of the Nineteenth Annual Symposium on Computational Geometry*, SCG '03, pages 312–321. ACM.

[Curl, 2000] Curl, J. S. (2000). *The Honourable the Irish Society and the Plantation of Ulster, 1608-2000: the City of London and the colonisation of County Londonderry in the Province of Ulster in Ireland: a history and critique*. Phillimore & Company.

[Davis et al., 2012] Davis, A., Levoy, M., and Durand, F. (2012). Unstructured light fields. *Computed Graphics Forum*, 31(2):305–314.

[Debevec, 1998] Debevec, P. (1998). Rendering synthetic objects into real scenes: Bridging traditional and image-based graphics with global illumination and high dynamic range photography. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '98, pages 189–198. ACM.

[Debevec et al., 1996] Debevec, P. E., Taylor, C. J., and Malik, J. (1996). Modeling and rendering architecture from photographs: a hybrid geometry- and image-based approach. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, SIGGRAPH '96, pages 11–20. ACM.

[Demmel et al., 1999] Demmel, J. W., Eisenstat, S. C., Gilbert, J. R., Li, X. S., and Liu, J. W. H. (1999). A supernodal approach to sparse partial pivoting. *SIAM Journal on Matrix Analysis and Applications*, 20(3):720–755.

[Denard, 2012] Denard, H. (2012). A new introduction to the london charter. In Bentkowska-Kafel, A., Baker, D., and Denard, H., editors, *Paradata and Transparency in Virtual Heritage*, pages 57–71. Ashgate.

[Dijkstra, 1959] Dijkstra, E. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271.

[Diringer, 1982] Diringer, D. (1982). *The Book Before Printing: Ancient, Medieval and Oriental*. Dover Publications.

[Easton Jr et al., 2003] Easton Jr, R. L., Knox, K. T., and Christens-Barry, W. A. (2003). Multispectral imaging of the archimedes palimpsest. In *Proceedings of 32nd Applied Imagery Pattern Recognition Workshop*, pages 111–116. IEEE.

[Endres, 2014] Endres, W. (2014). More than meets the eye: Going 3d with an early medieval manuscript. In Mills, C., Pidd, M., and Ward, E., editors, *Proceedings of the Digital Humanities Congress 2012*.

[Esteban and Schmitt, 2004] Esteban, C. H. and Schmitt, F. (2004). Silhouette and stereo fusion for 3d object modeling. *Computer Vision and Image Understanding*, 96(3):367–392.

[Federal Agencies Digitization Initiative, 2010] Federal Agencies Digitization Initiative (2010). *Technical Guidelines for Digitizing Cultural Heritage Materials: Creation of Raster Image Master Files*. Available at `http://www.digitizationguidelines.gov/guidelines/FADGI_Still_Image-Tech_Guidelines_2010-08-24.pdf`.

[Fischler and Bolles, 1981] Fischler, M. A. and Bolles, R. C. (1981). Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395.

[Fitzgibbon et al., 1998] Fitzgibbon, A., Cross, G., and Zisserman, A. (1998). Automatic 3d model construction for turn-table sequences. In *Proceedings of SMILE Workshop on Structure from Multiple Images in Large Scale Environments*, volume 1506 of *Lecture Notes in Computer Science*, pages 154–170. Springer Verlag.

[Furukawa and Ponce, 2010] Furukawa, Y. and Ponce, J. (2010). Accurate, dense, and robust multiview stereopsis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(8):1362–1376.

[Gortler et al., 1996] Gortler, S. J., Grzeszczuk, R., Szeliski, R., and Cohen, M. F. (1996). The lumigraph. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, SIGGRAPH '96, pages 43–54. ACM.

[Hanasusanto et al., 2010] Hanasusanto, G. A., Wu, Z., and Brown, M. S. (2010). Ink-bleed reduction using functional minimization. In *Proceedings of the 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 825–832. IEEE.

[Hartley and Zisserman, 2004] Hartley, R. and Zisserman, A. (2004). *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2nd edition.

[Hawkins et al., 2001] Hawkins, T., Cohen, J., and Debevec, P. (2001). A photometric approach to digitizing cultural artifacts. In *Proceedings of the 2001 Conference on Virtual Reality, Archeology, and Cultural Heritage*, VAST '01, pages 333–342. ACM.

[Heckbert, 1989] Heckbert, P. S. (1989). Fundamentals of texture mapping and image warping. Master's thesis, Computer Science Division, Electrical Engineering and Computer Science Department, University of California Berkeley.

[Heritage, 2011] Heritage, E. (2011). *3D Laser Scanning for Heritage: Advice and guidance to users on laser scanning in archaeology and architecture*. Available at `http://www.english-heritage.org.uk/publications/3d-laser-scanning-heritage2/3D_Laser_Scanning_final_low-res.pdf`.

[Hernandez Esteban and Schmitt, 2002] Hernandez Esteban, C. and Schmitt, F. (2002). Multi-stereo 3d object reconstruction. In *Proceedings of first International Symposium on 3D Data Processing Visualization and Transmission*, pages 159–166. IEEE.

[Horn and Johnson, 1986] Horn, R. A. and Johnson, C. R. (1986). *Matrix Analysis*, chapter 7. Cambridge University Press.

[Huang et al., 2010] Huang, Y., Brown, M. S., and Xu, D. (2010). User-assisted ink-bleed reduction. *IEEE Transactions on Image Processing*, 19(10):2646–2658.

[Iddan and Yahav, 2001] Iddan, G. J. and Yahav, G. (2001). Three-dimensional imaging in the studio and elsewhere. In *Proceedings of SPIE, Three-Dimensional Image Capture and Applications IV*, volume 4298, pages 48–55.

[Izadi et al., 2011] Izadi, S., Kim, D., Hilliges, O., Molyneaux, D., Newcombe, R., Kohli, P., Shotton, J., Hodges, S., Freeman, D., Davison, A., and Fitzgibbon, A. (2011). Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, UIST '11, pages 559–568.

[Kass et al., 1988] Kass, M., Witkin, A., and Terzopoulos, D. (1988). Snakes: Active contour models. *International Journal of Computer Vision*, 1:321–331.

[Kazhdan et al., 2006] Kazhdan, M., Bolitho, M., and Hoppe, H. (2006). Poisson surface reconstruction. In *Proceedings of the 4th Eurographics Symposium on Geometry Processing*, SGP '06, pages 61–70. Eurographics Association.

[Kim et al., 2010] Kim, S. J., Zhuo, S., Deng, F., Fu, C.-W., and Brown, M. (2010). Interactive visualization of hyperspectral images of historical documents. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1441–1448.

[Klein and Murray, 2007] Klein, G. and Murray, D. (2007). Parallel tracking and mapping for small ar workspaces. In *Proceedings of 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, ISMAR '07, pages 1–10.

[Klein et al., 2008] Klein, M. E., Aalderink, B. J., Padoan, R., Bruin, G. D., and Steemers, T. A. G. (2008). Quantitative hyperspectral reflectance imaging. *Sensors*, 8(9):5576–5618.

[Kobbelt, 2000] Kobbelt, L. (2000). $\sqrt{3}$-subdivision. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '00.

[Kruskal, 1964] Kruskal, J. B. (1964). Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika*, 29(1):1–27.

[Kumar et al., 2003] Kumar, S., Snyder, D., Duncan, D., Cohen, J., and Cooper, J. (2003). Digital preservation of ancient cuneiform tablets using 3d-scanning. In *Proceedings of IEEE Fourth International Conference on 3-D Digital Imaging and Modeling*, pages 326–333. IEEE.

[Levoy and Hanrahan, 1996] Levoy, M. and Hanrahan, P. (1996). Light field rendering. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, SIGGRAPH '96, pages 31–42. ACM.

[Levoy et al., 2000] Levoy, M., Pulli, K., Curless, B., Rusinkiewicz, S., Koller, D., Pereira, L., Ginzton, M., Anderson, S., Davis, J., Ginsberg, J., Shade, J., and Fulk, D. (2000). The digital michelangelo project: 3d scanning of large statues. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '00, pages 131–144. ACM.

[Lévy et al., 2002] Lévy, B., Petitjean, S., Ray, N., and Maillot, J. (2002). Least squares conformal maps for automatic texture atlas generation. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '02, pages 362–371. ACM.

[Library of Congress, 2007] Library of Congress (2007). *Technical Standards for Digital Conversion of Text and Graphic Materials*. Available at `http://www.digitizationguidelines.gov/guidelines/FADGI_Still_Image-Tech_Guidelines_2010-08-24.pdf`.

[London Metropolitan Archives, 2013] London Metropolitan Archives (2013). The great parchment book: Conserving, digitally reconstructing, transcribing, and publishing the manuscript known as the great parchment book. `http://www.greatparchmentbook.org/`.

[Lorensen and Cline, 1987] Lorensen, W. E. and Cline, H. E. (1987). Marching cubes: A high resolution 3d surface construction algorithm. In *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '87, pages 163–169. ACM.

[Lourakis and Argyros, 2009] Lourakis, M. I. and Argyros, A. A. (2009). Sba: A software package for generic sparse bundle adjustment. *ACM Transactions on Mathematical Software*, 36(1):1–30.

[Lowe, 2004] Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–110.

[Maver and Bajcsy, 1991] Maver, J. and Bajcsy, R. (1991). Occlusions as a guide for planning the next view. Technical report, University of Pennsylvania Department of Computer and Information Science.

[Mercer and Weiss, 2011] Mercer, P. and Weiss, V. (2011). *The New York State Capitol and the Great Fire of 1911*. Arcadia Publishing.

[Moody, 1939] Moody, T. W. (1939). *The Londonderry Plantation, 1609-41: The City of London and the Plantation in Ulster*. W. Mullan and Son.

[Moré, 1978] Moré, J. J. (1978). The levenberg-marquardt algorithm: implementation and theory. In *Numerical analysis*, pages 105–116. Springer.

[Newcombe and Davison, 2010] Newcombe, R. and Davison, A. (2010). Live dense reconstruction with a single moving camera. In *Proceedings of the 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1498–1505.

[Ohtake et al., 2003] Ohtake, Y., Belyaev, A., and Seidel, H.-P. (2003). A multi-scale approach to 3d scattered data interpolation with compactly supported basis functions. In *Proceedings of Shape Modeling International 2003*, pages 153–161.

[Otsu, 1975] Otsu, N. (1975). A threshold selection method from gray-level histograms. *Automatica*, 11(285-296):23–27.

[Praun et al., 2000] Praun, E., Finkelstein, A., and Hoppe, H. (2000). Lapped textures. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH 2000, pages 465–470. ACM.

[Ray et al., 2008] Ray, N., Vallet, B., Li, W. C., and Lévy, B. (2008). N-symmetry direction field design. *ACM Transactions on Graphics*, 27(2):10.

[Reed, 1975] Reed, R. (1975). *The Nature and Making of Parchment*. The Elmete Press.

[Reuter et al., 2009] Reuter, M., Biasotti, S., Giorgi, D., Patan, G., and Spagnuolo, M. (2009). Discrete laplacebeltrami operators for shape analysis and segmentation. *Computers and Graphics*, 33(3):381–390.

[Rusinkiewicz et al., 2002] Rusinkiewicz, S., Hall-Holt, O., and Levoy, M. (2002). Real-time 3d model acquisition. *ACM Transactions on Graphics*, 21:438–446.

[Samko et al., 2011] Samko, O., Lai, Y.-K., Marshall, D., and Rosin, P. (2011). Segmentation of parchment scrolls for virtual unrolling. In *Proceedings of the British Machine Vision Conference*, pages 37.1–37.11.

[Samko et al., 2014] Samko, O., Lai, Y.-K., Marshall, D., and Rosin, P. L. (2014). Virtual unrolling and information recovery from scanned scrolled historical documents. *Pattern Recognition*, 47(1):248–259.

[Schmitt and Yemez, 1999] Schmitt, F. and Yemez, Y. (1999). 3d color object reconstruction from 2d image sequences. In *Proceedings of the 1999 International Conference on Image Processing*, volume 3, pages 65–69.

[Schüller et al., 2013] Schüller, C., Kavan, L., Panozzo, D., and Sorkine-Hornung, O. (2013). Locally injective mappings. *Computer Graphics Forum (Proceedings of Eurographics/ACM SIGGRAPH Symposium on Geometry Processing)*, 32(5):125–135.

[Scott et al., 2003] Scott, W., Roth, G., and Rivest, J.-F. (2003). View planning for automated 3d object reconstruction inspection. *ACM Computing Surveys*, 35(1).

[Sheffer and de Sturler, 2001] Sheffer, A. and de Sturler, E. (2001). Parameterization of faceted surfaces for meshing using angle-based flattening. *Engineering with Computers*, 17(3):326–337.

[Sheffer et al., 2005] Sheffer, A., Lévy, B., Mogilnitsky, M., and Bogomyakov, A. (2005). Abf++: fast and robust angle based flattening. *ACM Transactions on Graphics*, 24(2):311–330.

[Shekhar et al., 1996] Shekhar, R., Fayyad, E., Yagel, R., and Cornhill, J. F. (1996). Octree-based decimation of marching cubes surfaces. In *Proceedings of Visualization*, pages 335–342. IEEE.

[Shewchuk, 2002] Shewchuk, J. R. (2002). Constrained delaunay tetrahedralizations and provably good boundary recovery. In *Proceedings of the Eleventh International Meshing Roundtable*, pages 193–204.

[Snavely et al., 2006] Snavely, N., Seitz, S. M., and Szeliski, R. (2006). Photo tourism: exploring photo collections in 3d. *ACM Transactions on Graphics*, 25(3):835–846.

[Snyder, 1993] Snyder, J. P. (1993). *Flattening the Earth : Two Thousand Years of Map Projections*. University of Chicago Press.

[Sorkine and Alexa, 2007] Sorkine, O. and Alexa, M. (2007). As-rigid-as-possible surface modeling. In *Proceedings of 2007 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, pages 109–116.

[Tenenbaum et al., 2000] Tenenbaum, J. B., Silva, V. D., and Langford, J. C. (2000). A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323.

[Terras, 2011] Terras, M. (2011). Artefacts and errors: Acknowledging issues of representation in the digital imaging of ancient texts. In Fischer, F., Fritze, C., and Vogeler, G., editors, *Kodikologie und Paläographie im digitalen Zeitalter 2 / Codicology and Palaeography in the Digital Age 2*, pages 43–61. Books on Demand: Norderstedt.

[Thomson and Kite, 2006] Thomson, R. and Kite, M. (2006). *Conservation of Leather and Related Materials*. Elsevier Butterworth-Heinemann.

[Tian and Narasimhan, 2011] Tian, Y. and Narasimhan, S. G. (2011). Rectification and 3d reconstruction of curved document images. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR),*, pages 377–384.

[Tite, 1994] Tite, C. G. C. (1994). *The Manuscript Library of Sir Robert Cotton*. The British Library.

[Tite, 2003] Tite, C. G. C. (2003). *The Early Records of Sir Robert Cotton's Library: Formation, Cataloguing, Use*. The British Library.

[Tomalak, 2013] Tomalak, A. (2013). Crisp as a poppadom. `http://britishlibrary.typepad.co.uk/digitisedmanuscripts/2013/02/crisp-as-a-poppadom.html`.

[Ulges et al., 2004] Ulges, A., Lampert, C. H., and Breuel, T. (2004). Document capture using stereo vision. In *Proceedings of the 2004 ACM Symposium on Document Engineering*, DocEng '04, pages 198–200. ACM.

[V. de Silva, 2004] V. de Silva, J. T. (2004). Sparse multidimensional scaling using landmark points. Technical report, Stanford University.

[Visual Computing Lab of CNR-ISTI, 2005] Visual Computing Lab of CNR-ISTI (2005). Meshlab. `http://meshlab.sourceforge.net/`.

[Weickert, 1999] Weickert, J. (1999). Coherence-enhancing diffusion filtering. *International Journal of Computer Vision*, 31(2-3):111 – 127.

[Westermann et al., 1999] Westermann, R., Kobbelt, L., and Ertl, T. (1999). Real-time exploration of regular volume data by adaptive reconstruction of isosurfaces. *The Visual Computer*, 15(2):100–111.

[Wilhelms and Van Gelder, 1992] Wilhelms, J. and Van Gelder, A. (1992). Octrees for faster isosurface generation. *ACM Transactions on Graphics (TOG)*, 11(3):201–227.

[Woods, 1995] Woods, C. (1995). Conservation treatments for parchment documents. *Journal of the Society of Archivists*, 16(2):221–238.

[Wu, 2007] Wu, C. (2007). Siftgpu: A gpu implementation of scale invariant feature transform (sift). `http://cs.unc.edu/~ccwu/siftgpu`.

[Wu, 2011] Wu, C. (2011). Visualsfm: A visual structure from motion system. `http://www.cs.washington.edu/homes/ccwu/vsfm/`.

[Wu and Agam, 2002] Wu, C. and Agam, G. (2002). Document image dewarping for text/graphics recognition. In *Proceedings of the Joint IAPR International Workshop on Structural, Syntactic, and Statistical Pattern Recognition*, pages 348–357.

[Wu et al., 2011] Wu, C., Agarwal, S., Curless, B., and Seitz, S. M. (2011). Multicore bundle adjustment. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3057–3064.

[Xu and Prince, 1998] Xu, C. and Prince, J. L. (1998). Snakes, shapes, and gradient vector flow. *IEEE Transactions on Image Processing*, 7(3):359–369.

[Yamashita et al., 2004] Yamashita, A., Kawarago, A., Kaneko, T., and Miura, K. T. (2004). Shape reconstruction and image restoration for non-flat surfaces of documents with a stereo vision system. In *Proceedings of the 17th International Conference on Pattern Recognition*, ICPR '04, pages 482–485. IEEE.

[Youtie, 1963] Youtie, H. C. (1963). The papyrologist: Artificer of fact. *Greek, Roman, and Byzantine Studies*, 4(1):19–32.